commodore

16

Felhasználói kézikönyv

COMMODORE 16

Felhasználói kézikönyv

Fordította és szerkesztette:
TORY KÁLMÁN

Lektorálta:
DR. VARGA ANDRÁS
KUTASSY LÁSZLÓ

Ez a könyv a szerzői joggal védett COMMODORE 16 USER MANUAL című kiadványnak a Commodore Electronics Limited engedélye alapján készült fordítása.
A kiadványt az 1969. évi III. törvény 13. § és a 18. § első bekezdése alapján tilos bármilyen módon sokszorosítani és a sokszorosítást terjeszteni.

A kiadásért felel a NOVOTRADE RT. igazgatója.
Kiadványmenedzser: BÉKÉS TAMÁS
Műszaki szerkesztő: DÉVÉNYI ERIKA
ISBN: 96302 3672 9

TARTALOMJEGYZÉK

1. FEJEZET Összeállítás 5
2. FEJEZET A billentyűzet 15
3. FEJEZET Szoftver 25
4. FEJEZET Az első lépések 35
5. FEJEZET Számok és műveletek 45
6. FEJEZET Grafika és szín 55
7. FEJEZET Hang és zene 73

BASIC 3.5 ISMERETTÁR

Parancsok 82
Utasítások 94
Függvények 120
Változók és operátorok 127
Rövidítések és hivatkozási táblázat 131

FÜGGELÉKEK

Hibaüzenetek 136
Lemez hibaüzenetek 138
Matematikai függvények származtatása 142
Zenei hangjegytáblázat 143
Képernyőkódok 144
ASCII és CHR$ kódok 147
Irodalomjegyzék 149

TÁRGYMUTATÓ 151

KIEGÉSZÍTÉS

a Tudományszervezési és Informatikai Intézet által forgalmazott, általános iskolai célra szánt **magyar ékezetes betűkkel ellátott** COMMODORE 16 számítógép Felhasználói Kézikönyvéhez

Általános leírás: A magyar ékezetes betűk csak kisbetű/nagybetű (v. írógép) üzemmódban használhatók. A következő 18 billentyűn lévő, az említett üzemmódban a C= (Commodore) billentyűvel együttesen használt baloldali grafikus karakterek helyett jelennek meg:

Billentyű:	Q	W	E	R	T	Y	U	I	O	A	S	D	F	G	H	J	K	L
Karakter:	Á	Ü	É	Ű	Ő	Ö	Ú	Í	Ó	á	ü	é	ű	ő	ö	ú	í	ó

A billentyűzeten a grafikus karakterek mellett az ékezetes betűk fel vannak tüntetve.

Módosítások a Felhasználói Kézikönyvben:

A 2. fejezetben, a 16. oldalon ismertetésre került az írógép üzemmód. A magyar ékezetes nagy- és kisbetűk csakis ebben az üzemmódban jeleníthetők meg, mégpedig a baloldali grafikus karakterek helyett. A 16. oldal alján lévő példa esetén a C= + A billentyűk lenyomása esetén nem a ┌ grafikus jel, hanem az Á betű jelenik meg.

A 19. oldalon, a COMMODORE BILLENTYŰ-ről írtak így módosulnak:

2. A C= billentyű lehetővé teszi a billentyűk előlapjának baloldalán található grafikus jelek, ill. **írógép üzemmódban (18 billentyűn) az ugyanott található magyar ékezetes nagy- és kisbetűk** bevitelét.

A 21. oldalon, a "Grafikus billentyűk" szakasz első mondata így módosul:

Minden betű billentyű (és néhány egyéb billentyű) előlapján két különböző jel, **18 betű billentyű előlapján pedig harmadik jelként egy-egy ékezetes kis- vagy nagy betű** látható.

A szakasz utolsó előtti mondata a **22. oldalon** pedig eképpen: Írógép üzemmódban csak a billentyű bal oldalán lévő grafikus jelet, ill. **ha ékezetes betű van rajta, akkor csakis azt** használjuk (a C= és a megfelelő billentyű lenyomásával).

Az 56. oldalon, a 11. sor mondata így módosul:

Ha a C= billentyűt egy betű billentyűvel együtt nyomjuk meg, akkor **nem írógép üzemmódban** a betű baloldalán látható grafikus jelet kapjuk, **írógép üzemmódban a csak két grafikus jellel ellátott betűk esetén ugyancsak a baloldali grafikus jelet, az ékezetes betűvel is ellátott 18 betű esetén pedig az ékezetes betűt kapjuk.**

1. FEJEZET

Üzembehelyezés

- A Commodore 16 kicsomagolása
- Ismerkedés a kapcsolókkal és a csatlakozókkal
- A Commodore 16 üzembehelyezése
- Hibakereső táblázat
- Perifériák

A COMMODORE 16 KICSOMAGOLÁSA

Most, hogy kinyitotta az új Commodore 16 gépet tartalmazó dobozt és megtalálta ezt a kézikönyvet, első dolga legyen ellenőrizni, hogy az alábbi lista szerint minden megvan-e.

1. A Commodore 16
2. A tápegység
3. A TV csatlakozókábel
4. A felhasználói kézikönyv (Feltehetőleg megtalálta, mivel épp ezt olvassa)
5. Egyéb: Garancialevél

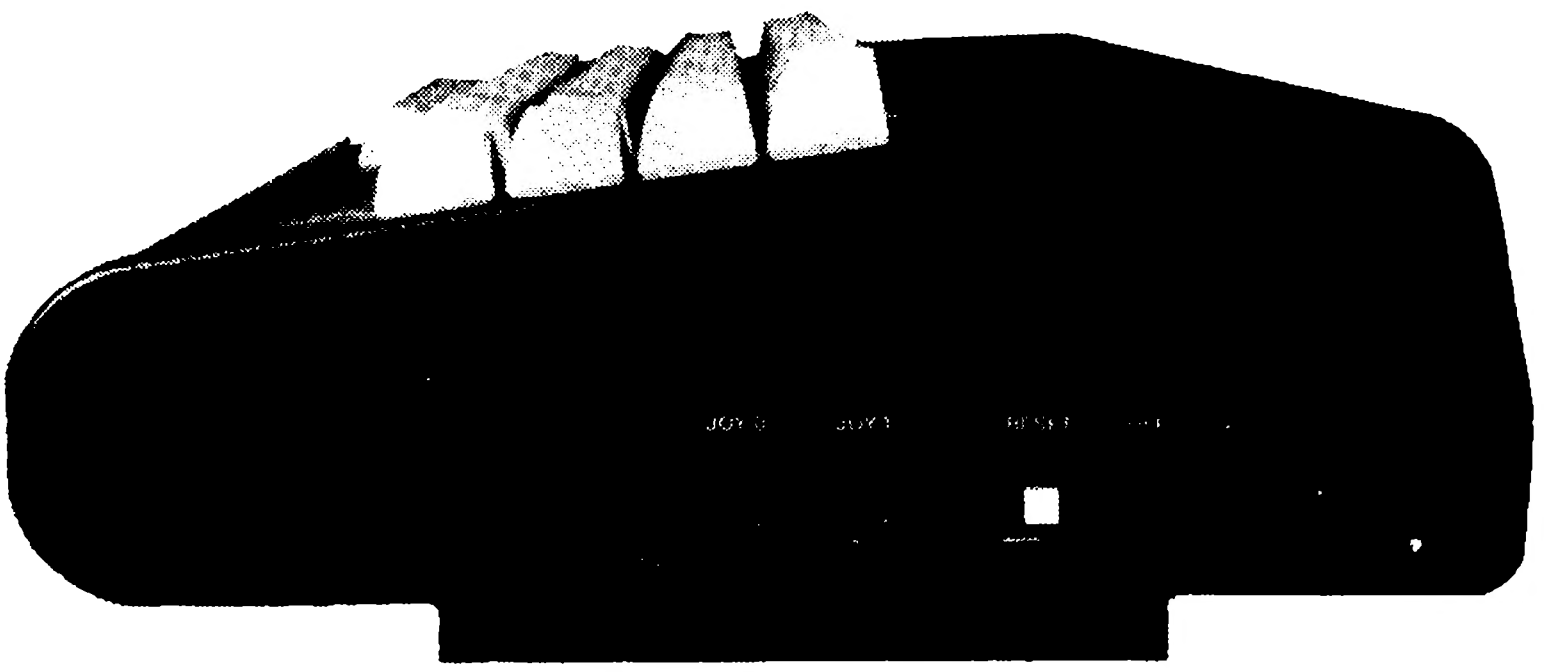
Ha nem találta meg mindezeket a dobozban, kérje pótlásukat az eladótól. Mielőtt bármit csatlakoztatna, nézze át a következő képeket. Megismerkedhet a kivezetésekkel és kapcsolókkal, így a számítógéprendszert gyorsan és könnyen össze tudja állítani.

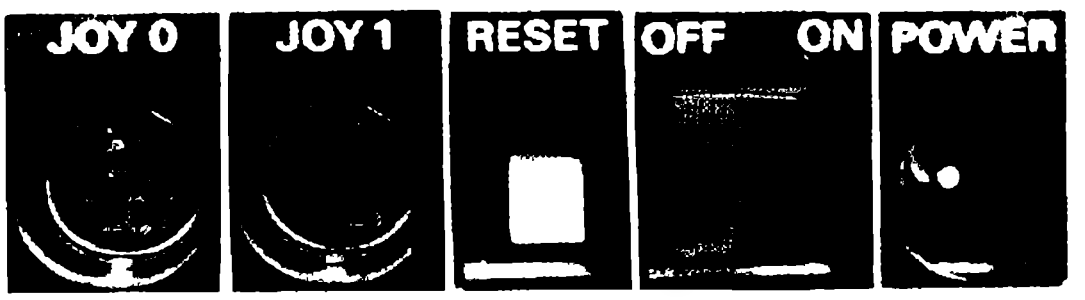
A KAPCSOLÓK ÉS CSATLAKOZÓK ISMERTETÉSE

A Commodore 16 jobb oldala

ÜZEMI KAPCSOLÓ (ON/OFF)

Az ON/OFF feliratú kapcsoló szolgál a készülék KI-BE kapcsolására. A Commodore 16 számítógépet ki kell kapcsolni (OFF), mielőtt csatlakoztatja vagy eltávolítja a bővítéseket vagy egyéb perifériákat, pl. a nyomtatót vagy a kazettás magnót.





Jobbra a billentyűzet fölött piros lámpa található, mely a tápfeszültség bekapcsolását jelzi.

A BOTKORMÁNY CSATLAKOZÓK

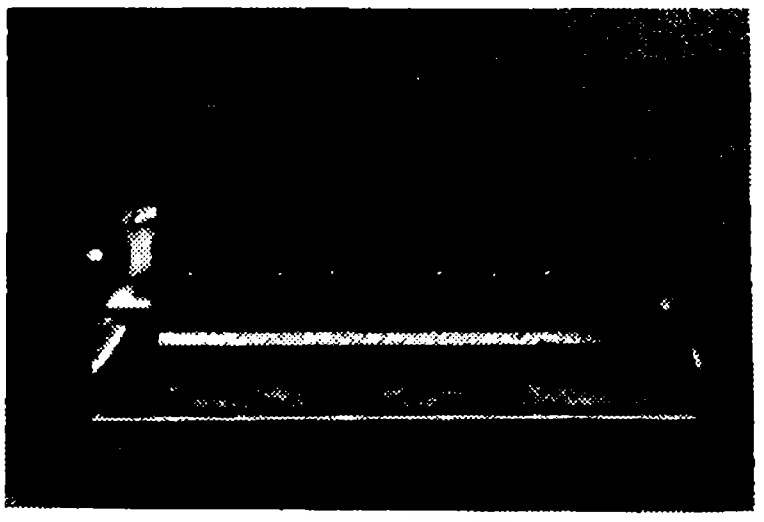
A JOY 1 és JOY 2 jelű csatlakozóaljzatba botkormányokat lehet csatlakoztatni.

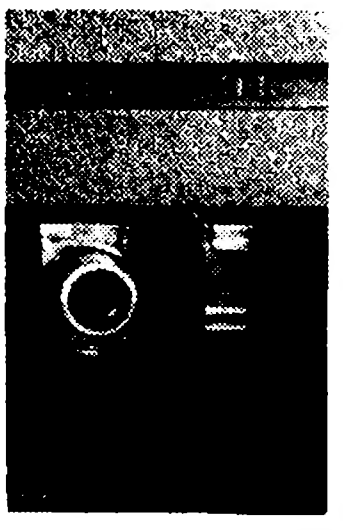
A RESET GOMB

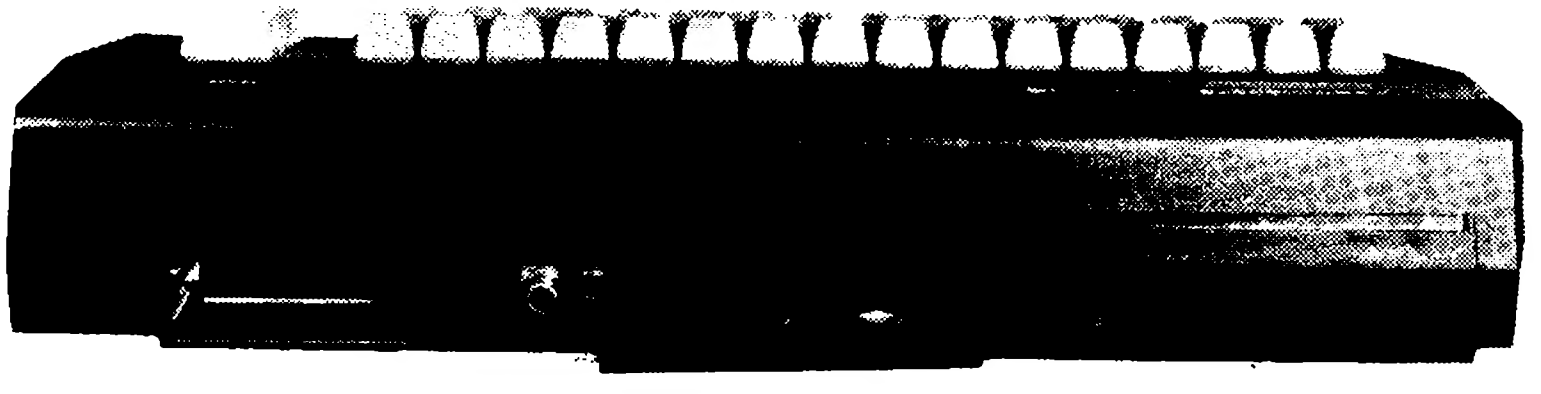
Akkor nyomja meg a RESET gombot, ha úgy akarja újraindítani a gépet, mintha éppen bekapcsolta volna. A RESET gomb törli a képernyőt és a begépelt BASIC programokat. Letörölhetjük a képernyőt úgy is, hogy a programokat nem töröljük. Ezt a 4. fejezetben tárgyaljuk.

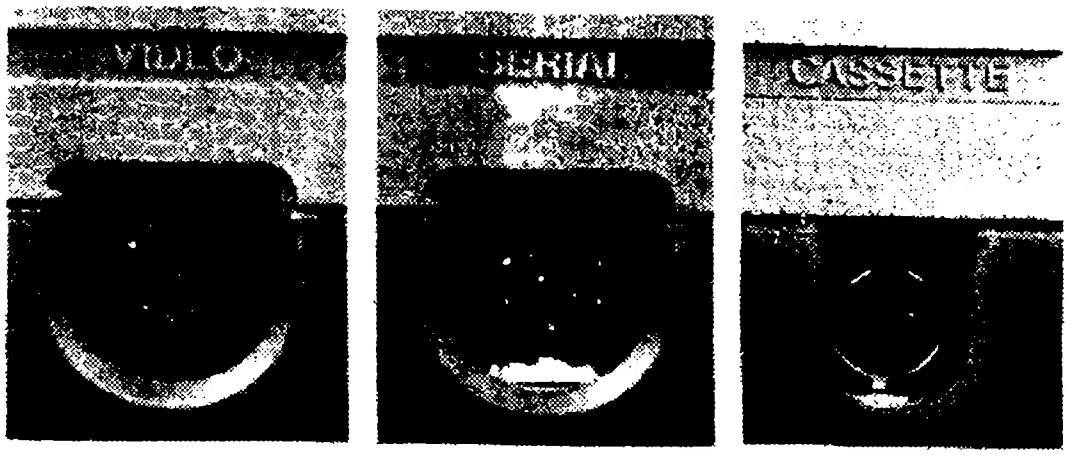
TÁPEGYSÉG CSATLAKOZÓ (POWER)

A POWER feliratú csatlakozó a tápegység csatlakoztatására szolgál. A készülékhez adott tápegység 9,5 V-os kimenetét – a koncentrikus csatlakozódugót – kell ide csatlakoztatni. EZUTÁN csatlakoztathatjuk a tápegység hálózati csatlakozó dugóját a fali dugaszolóaljzatba.









A MEMÓRIABŐVÍTŐ KAPU (MEMORY EXPANSION)

A Commodore 16 programmodulokat dugjuk ide. Mielőtt behelyezzük vagy eltávolítjuk a modulokat, bizonyosodjunk meg arról, hogy a gép ki van-e kapcsolva (Off).

A TV-CSATLAKOZÓ

Ide kell dugni a TV-kábel (a vékony fekete kábel) egyik végét. Csak az egyik vége illik ebbe a csatlakozóba, a másik a TV-be való.

A VIDEO CSATLAKOZÓ ALJZAT

Ide dugjuk azt a kábelt, mely monitort csatlakoztat a Commodore 16-hoz. Ha TV készüléket kapcsolunk a számítógéphez, akkor nem kell ezt használni.

SOROS CSATLAKOZÓ ALJZAT (SERIAL)

Ebbe az aljzatba lemezmeghajtót vagy sornyomtatót csatlakoztathatunk. Ha mindkettőt használni akarjuk, akkor először a lemezmeghajtót dugjuk ebbe a kivezetésbe, a nyomtató kábelét pedig a lemezmeghajtó hátlapján lévő csatlakozóba.

A KAZETTA KAPU (CASSETTE)

A Commodore 1531 kazetta egységet dugaszolhatjuk ide.

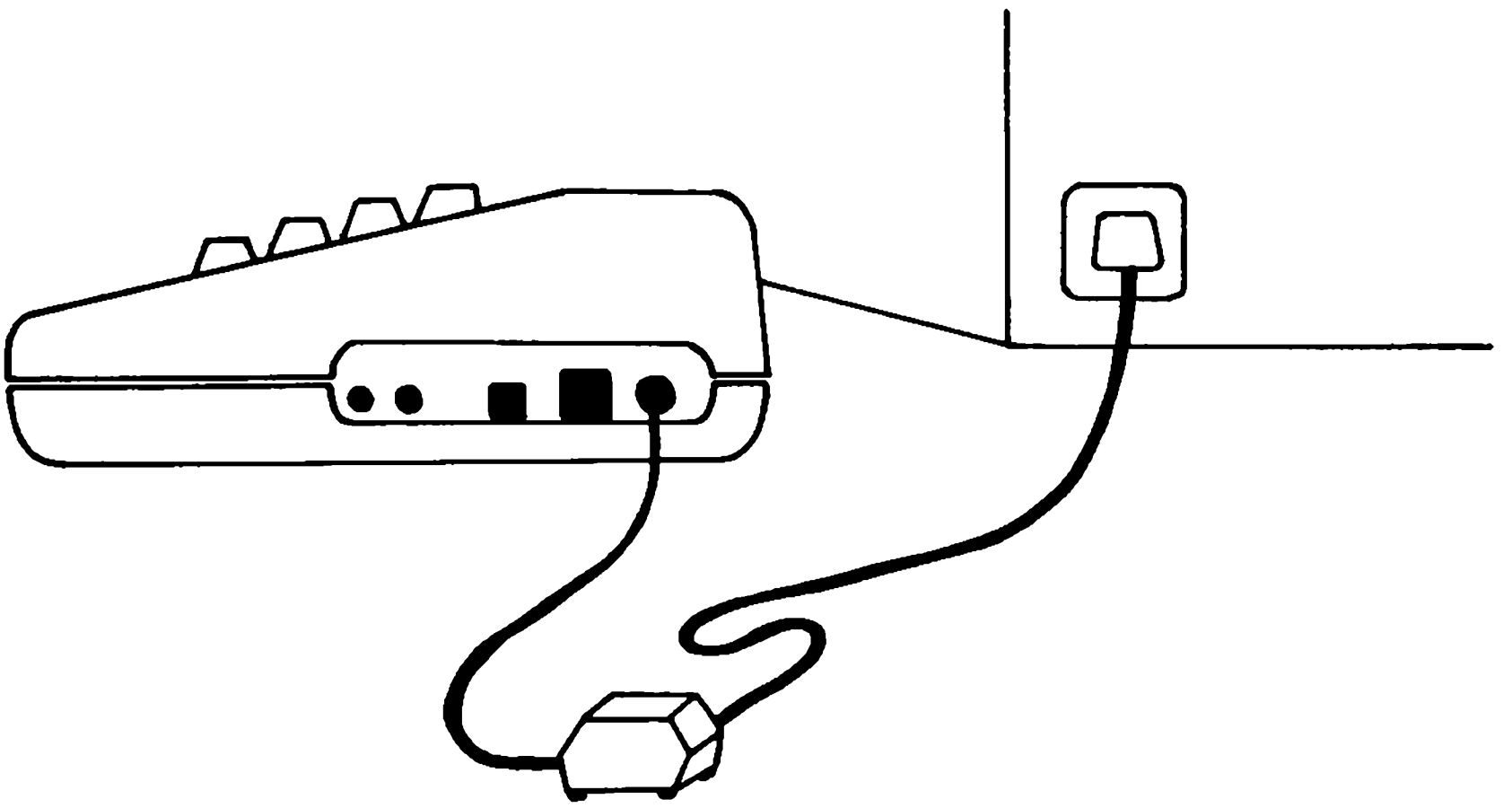
A Commodore 16 összeállítása

A Commodore 16 csatlakoztatása igen egyszerű! Csak a következőket kell tennie:

Győződjön meg arról, hogy elegendő fali elektromos dugaszoló aljzat áll rendelkezésére! Lehet, hogy elosztóra vagy hosszabbítóra lesz szüksége, különösen ha nyomtatót vagy lemezmeghajtót is csatlakoztatni akar. Emlékezzen arra, hogy minden berendezésnek (TV, számítógép, stb.) kikapcsolt állapotban kell lennie, amíg teljesen össze nem állította a gépet.

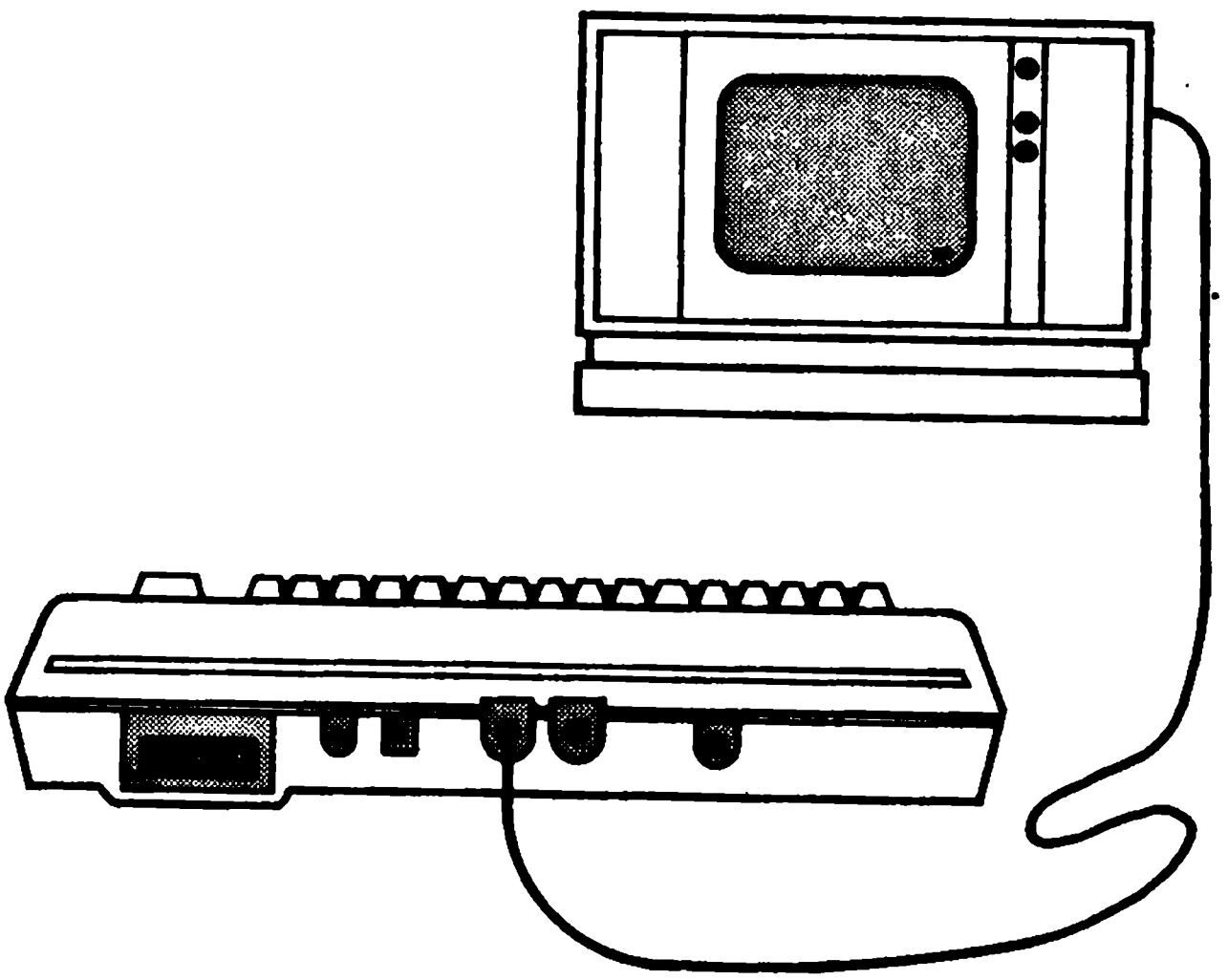
1. A TÁPEGYSÉG CSATLAKOZTATÁSA

- Először dugja a tápegység koncentrikus kimeneti csatlakozóját a számítógép jobb oldalán lévő POWER feliratú aljzatba!
- Azután dugja a tápegység hálózati csatlakozóját a fali hálózati aljzatba!



2. A TV-KÁBEL CSATLAKOZTATÁSA

- Dugja a TV-kábel egyik végét (csak az egyik illik bele) a TV antenna-csatlakozójába!
- A kábel másik végét dugja a számítógép hátlapján lévő TV-csatlakozó aljzatba!



A COMMODORE 16 CSATLAKOZTATÁSA MONITORHOZ

Ha a számítógépet TV helyett monitorhoz csatlakoztatjuk, kövessük a monitorhoz tartozó kézikönyv utasításait. Csak egy kábelt kell csatlakoztatni, ami közvetlenül.a monitortól a számítógép hátlapján lévő video csatlakozóhoz megy.

VÉGÜL...

Itt az ideje, hogy bekapcsoljuk a számítógépet (ha figyelmesen állította össze, akkor már tudja hol az üzemi kapcsoló). Ha minden rendben ment, a piros lámpa kigyullad, és a következő üzenet jelenik meg a képernyőn:

COMMODORE BASIC V3.5 12277 BYTES FREE
READY.
■

A Ready üzenet alatt villogó kurzor azt jelzi, hogy a Commodore 16 beírásra vár. A háttér színe fehér, míg a gépelt betűk feketék, világoskék kerettel a képernyő körül.
Ha nem ment minden rendben, jó hasznát veszi a hibakereső táblázatnak.

HIBAKERESŐ TÁBLÁZAT

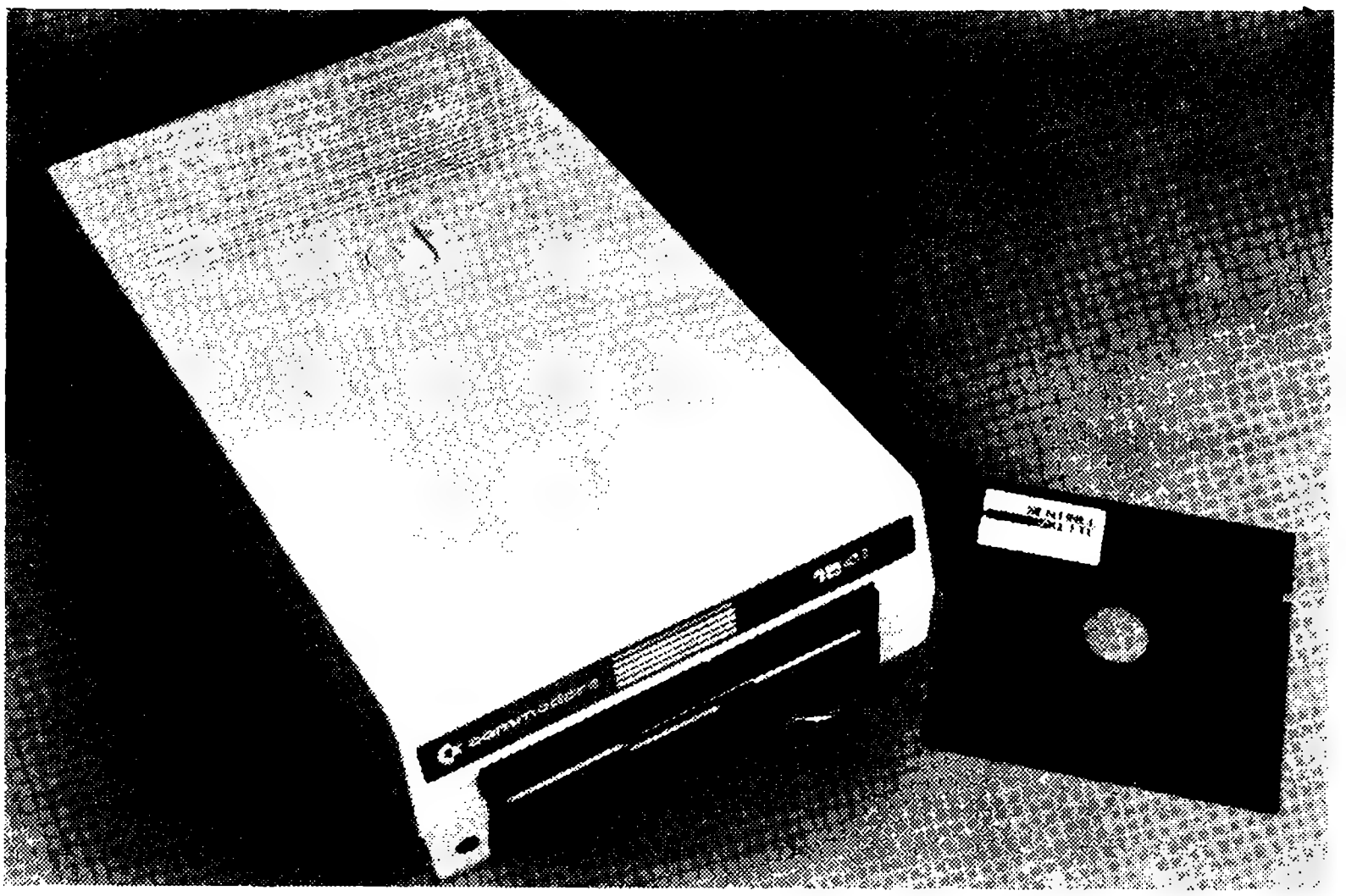
Tünet	Ok	Teendő
A jelzőlámpa nem világít	A számítógép nincs bekapcsolva	Győződjünk meg arról, hogy az üzemi kapcsoló ON állásban van-e
	A tápegység csatlakozó nincs bedugva	Ellenőrizzük, hogy a POWER feliratú aljzat nem laza-e és nincs-e kihúzva a tápegység csatlakozó
	A tápegységet nem csatlakoztattuk a hálózatba	Ellenőrizzük a csatlakozást a fali csatlakozónál
	Rossz a biztosíték a tápegységben	Szervizzel cseréltessük ki
Nincs kép	A TV-kábel nincs bedugva	Ellenőrizzük a TV-kábel csatlakozását
	A TV nincs bekapcsolva	Kapcsoljuk be a TV-t
Kuszа kép a képernyőn a modul csatlakoztatása után	A modul nincs megfelelően betéve	Helyezzük be újra a modult a gép kikapcsolása után
A kép nem színes	A TV rosszul van beállítva	Hangoljuk újra a TV-t
	A TV nincs megfelelően csatlakoztatva	Ellenőrizzük a csatlakoztatásokat
	A színtelítettség rosszul van beállítva a gépen vagy a TV-n	Állítsuk be a színeket
A kép rendben, hang nincs	A TV hangereje túl kicsi	Állítsuk be a TV hangerejét
	A TV rosszul van beállítva	Hangoljuk újra a TV-t.

FONTOS: Némely TV-készülék nem képes a teljes Commodore 16 képernyő kijelzésére. Ezek a TV-k levágják a két szélső oszlopot a képernyőről. Javasoljuk más TV készülék vagy monitor, pl. a Commodore 1702, 1802 vagy 1803 színes monitorok használatát.
Ha ez nem lehetséges, az ESC majd az R billentyűk lenyomásával oldhatjuk meg a problémát. Ezzel a számítógép képernyőmérete 38 oszlopra csökken, így a teljes kép elfér a képernyőn. Ezt minden bekapcsolás vagy Reset után meg kell ismételni.

Csak a kétnormás (vagyis SECAM és PAL) tv-készülék alkalmazható (ha csak a Magyarországon használt SECAM rendszert tudja, akkor a kép csak fekete-fehér lesz, és nem lesz hang).

Perifériák

A perifériák olyan berendezések, melyek a Commodore 16-hoz csatlakoztatva kiterjesztik a gép alkalmazhatóságát. Ezek a tartozékok kiterjesztik a számítógép kihasználását. A perifériák teszik lehetővé, hogy adatokat tároljunk vagy őrizzünk meg, az ernyőn megjelenő képet kinyomtassuk (fekete-fehérben vagy színesben), kazettán vagy diszketten tárolt programokat használjunk, és a kijelzésre szánt adatokról tiszta, éles képet kapjunk.



Programok megőrzéséhez és betöltéséhez adattároló eszközre van szükségünk. Adatokat rögzíthetünk mágneskazettára és lemezre, és ezekről az adathordozókról vissza is tölthetjük azokat. Kazettán tárolt programok alkalmazásához (és saját programjaink kazettára vételéhez) Commodore 1531 DATASSETTE magnetofonra van szükségünk. Mágneslemezekhez különböző mágneslemez meghajtóegységeket használhatunk. Ezek különösen gyorsak, és hatékonyan alkalmazhatók. A Commodore 16-tal kompatibilis meghajtók a Commodore 1541 és 1551.
Lehet, hogy a TV-készülék nem ad olyan tiszta képet, amilyet a számítógéptől elvárnánk. A Commodore színes monitorokat speciálisan arra tervezték, hogy a számítógép kimenetéről a legélesebb és legfényesebb képet adják. A TV-készüléket a video-bemenetről monitorként használva élesebb és tisztább képet kapunk.

Ha szövegszerkesztő programot vagy grafikus programcsomagot használunk a számítógépen, a nyomtatók papíron reprodukálják azt, ami a képernyőn megjelenik. Többféle nyomtató kapható, ami számítógépünkhöz illeszthető. Ilyenek például az MPS–801, MPS–802, MPS–803 (írógép-papírral is működik) és a DPS 1101 (betűminősége levélíráshoz is megfelelő).

Műszaki adatok

Alapgép

Tápfeszültség: 9,5 V (csak saját tápegységével üzemeltethető)
Teljesítmény-felvétel: max. 7,6 W
Tár: 32 Kbájt ROM (Operációs rendszer és BASIC 3.5 interpreter)
16 Kbájt RAM

Processzor:	7501 mikroprocesszor 1,6 MHz órajellel
Grafika:	40 oszlop x 25 sor szöveg
	15 szín 8 fényerősséggel + fekete
	Nagy- és kisbetűk, grafikus jelek
	Finomgrafika (320x200 pixel) teljes BASIC támogatással
	Osztott képernyő (szöveg és grafika keverve)
Hang:	2 hanggenerátor, egyik zajgenerátorként is használható
	8 hangerősségi szint
Billentyűzet:	66 billentyű, 4 programozható funkció billentyű, 4 kurzor billentyű
Be/Kimenet	Soros interfész (lemez, nyomtató csatlakoztatáshoz)
	Kazetta egység csatlakozó
	Modul csatlakozó
	2 botkormány csatlakozó
	Video kimenet (PAL kompozit jel)
	Hang be- és kimenet (a video csatlakozón)
	TV kimenet (~ 36. csatorna, PAL rendszerű)
Méret:	40x8x20,5 cm
Tömeg:	1750 g

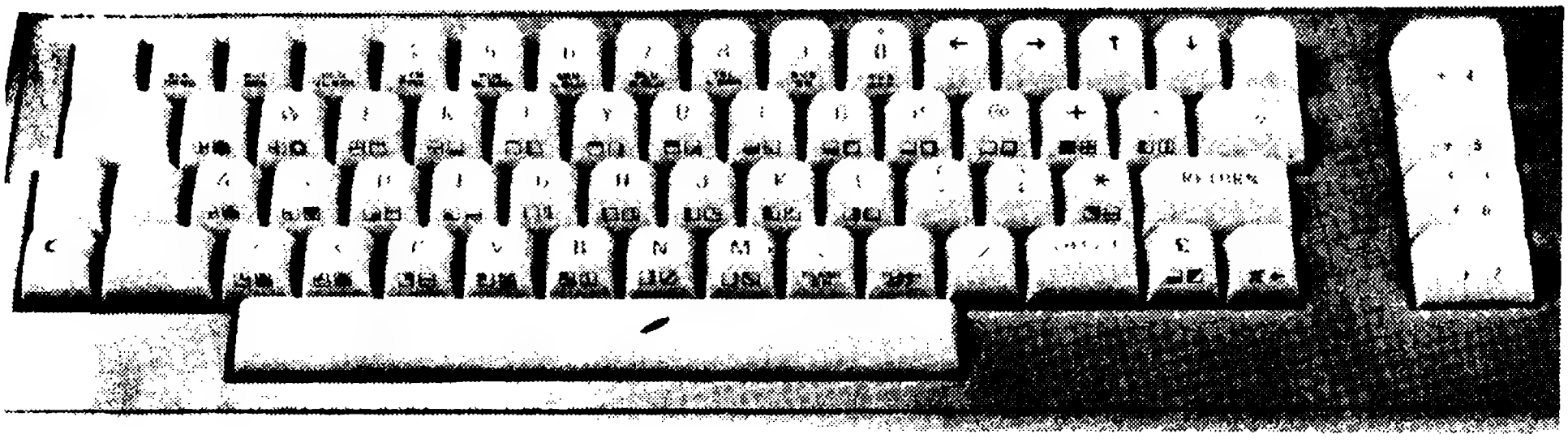
Tápegység

Alkalmazott érintésvédelmi mód: II.
Megerősített (kettős) szigetelésű készülék
Névleges hálózati feszültség: 220 V ± 10% – 15%, 50 Hz
Teljesítmény-felvétel: max. 15 W
Kimeneti stabilizált feszültség: 9,5 V, max. 800 mA egyenáram
Kimeneti teljesítmény: max. 7,6 W

2. FEJEZET

A billentyűzet

- A billentyűzetről általában
- Speciális billentyűk
- Grafikus billentyűk
- Funkció billentyűk
- A HELP billentyű



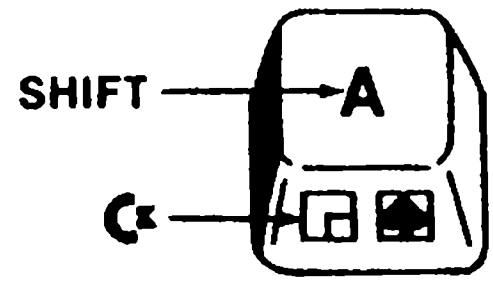
A BILLENTYŰZETRŐL ÁLTALÁBAN

A Commodore 16 billentyűzetén lévő billentyűk többsége az írógép billentyűivel azonos, de minden billentyű több feladatot lát el, mint az írógépen. Ebben a részben megismerkedünk a különleges billentyűk, mint például a C= billentyű és a négy különálló kurzor billentyű használatával. Látni fogjuk minden billentyű különleges alkalmazását, és többek között azt is, hogyan használjuk a legtöbb billentyű előlapján található grafikus jeleket. A billentyűk megismeréséhez célszerű azokat megkeresni és használatukat gyakorolni.

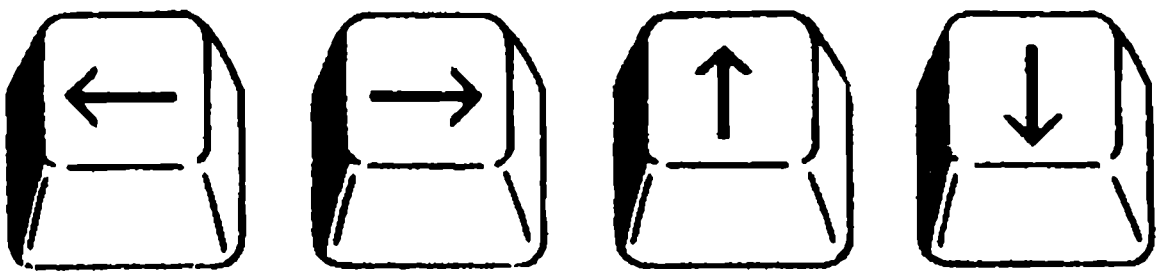
A BILLENTYŰZET ÍRÓGÉPKÉNT VALÓ HASZNÁLATA

Ha leütünk egy betűt a számítógépen, az nagybetűként jelenik meg a képernyőn. Ha a billentyűt egymagában ütjük le, a betűk és számok pontosan úgy jelennek meg a képernyőn, ahogy azok a billentyűkön láthatók. Több egyéb billentyűt (+ , − , = , @, * és az angol font jel) egymagában kell lenyomni, de a legtöbb írásjelet a SHIFT billentyűvel együtt. Ha "szabályosan" gépelni akarunk, nagy- és kisbetűket is írhatunk (mint az írógépen) a SHIFT és C= billentyűk egyidejű lenyomása után. Ilyenkor az összes betűbillentyű kisbetűt ír ki. Ha SHIFT billentyűt egy betűvel együtt nyomjuk meg, nagybetűt kapunk. A számok és írásjelek ugyanúgy működnek, mint "nem írógép" üzemmódban. A két billentyű (SHIFT és C=) együttes lenyomása után újra megszűnik az írógép üzemmód.

Billentyű	+ **Billentyű**	=
SHIFT	+ C=	= Írógép üzemmódba belépés vagy kilépés belőle
A		= a
SHIFT	+ A	= A
C=	+ A	= ┌



Különleges billentyűk



A Commodore 16 billentyűzetén néhány billentyű egészen másként viselkedik, mint ahogy bármely jólnevelt írógépbillentyűről feltételezhetnénk. Ezek a billentyűk vagy olyan feladatok elvégzésére tesznek alkalmassá más billentyűket, amelyekre azok különben nem képesek, vagy a számítógép működésével állnak kapcsolatban. A Commodore 16 olyan különleges jeleket is tartalmaz, amelyek nem találhatók meg a legtöbb írógépen, sőt a legtöbb számítógépen sem. Ezek a különleges jelek az angol font (£), a Pi (π), a kisebb és nagyobb jelek ($<$ $>$), a szögletes zárójelek ([ ]), és nyilak ($\leftarrow$ $\uparrow$). Ezeket a különleges jeleket gyakran használjuk programírás közben.

RETURN

A RETURN billentyűt minden a Commodore 16 billentyűzetén beírt utasítássor végén le kell ütni. Úgy működik, mint egy ENTER (bevitel) billentyű, mivel a RETURN utasításokat és adatokat visz be a számítógépbe.

SHIFT

Már láttunk egy példát a SHIFT billentyű alkalmazására, mikor a billentyűzetet írógépként használtuk. Ez jellemző a SHIFT billentyűre: mindig csak más billentyűk működését módosítja, de önmagában semmit sem csinál. (Mindig csak koszorúslány, soha nem menyasszony...). A SHIFT billentyű teszi lehetővé, hogy nagybetűket, grafikus jeleket, és írásjeleket írjunk, és néhány egyéb apróságot is lehetővé tesz — egy másik billentyű segítségével.

A SHIFT LOCK billentyű hatása ugyanaz, mint a SHIFT billentyűé, de rögzíti annak helyzetét, hogy ne kelljen állandóan lenyomva tartani. Ha a SHIFT LOCK billentyű le van nyomva, minden karaktert betűváltóval gépelünk be. A SHIFT LOCK billentyű ismételt lenyomása kioldja a betűváltót, és az alapállapot tér vissza.

RUN/STOP BILLENTYŰ

Ha ezt a billentyűt lenyomjuk, megállítjuk a Commodore 16-on futó BASIC programot. Ha a billentyűt programfutás közben nyomjuk le, a billentyűzetnek adjuk át a vezérlést.

A SHIFT és a RUN/STOP billentyűk egyidejű lenyomásával a Commodore 16 a lemezmeghajtó egységben lévő lemezen található első programot tölti be és indítja el.

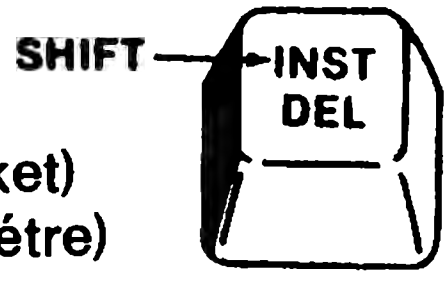
A kurzor billentyűk

A kurzor egy villogó jel, ami a képernyőn az első szabad karakter helyét jelzi; gyorsan és könnyen mozgatható a kurzor billentyűk segítségével. Négy különálló kurzor billentyű van, mindegyiken nyíl mutatja az irányt, amerre a billentyű a kurzort mozgatja: balra, jobbra, fel vagy le. A billentyűkkel a kurzort a képernyőn lévő bármely szöveg fölött mozgathatjuk anélkül, hogy a már beírt karakterekre bármilyen hatással lenne. A Commodore 16 billentyűzetén lévő összes billentyűhöz hasonlóan minden kurzor billentyű automatikusan ismételhető. Bármely billentyűt folyamatosan lenyomva tartva a kurzor folyamatosan a billentyű által meghatározott irányba mozog, (megkímélve minket attól, hogy a mesebeli "gyorstüzelőt" kelljen utánoznunk).

INST/DEL (BESZÚR, TÖRÖL)

Ezzel a billentyűvel betűket és számokat toldhatunk be és törölhetünk abban a sorban, amelyben a kurzor van. Ha a billentyűt magában ütjük le, a kurzor bal oldalán található karakter törlődik és a kurzor a karakter helyére lép. A kurzor billentyű arra is alkalmas, hogy a sor közepére visszalépve a DEL billentyű segítségével kitöröljünk egy betűt. Ilyenkor a kurzor bal oldalán lévő betű törlődik, és a sor többi része eggyel balra lép, kitöltve a keletkezett hézagot.
Ha betűk és számok betoldásához helyet akarunk csinálni, a billentyűt a SHIFT-tel együtt kell lenyomnunk. Ilyenkor a kurzor jobb oldalán szóköz keletkezik, míg a kurzor maga nem mozdul. Ha a szóközt egy sor közepére toldjuk be, a sor többi része jobbra mozdul el.
Az INST/DEL billentyűvel sok időt takaríthatunk meg, ha a már beírt szöveget szerkeszteni vagy módosítani akarjuk.

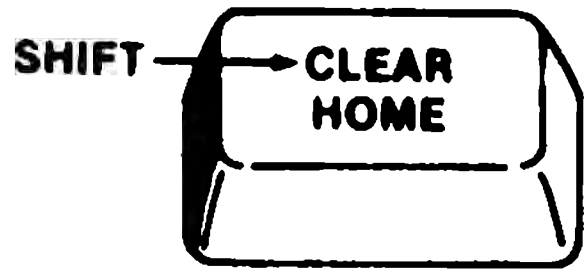
Billentyű	Hatás
INST/DEL	= Törlés (törli a beírt karaktereket)
SHIFT + INST/DEL	= Beszúrás (üres helyeket hoz létre)



CLEAR/HOME (KÉPERNYŐTÖRLÉS, KEZDETI POZÍCIÓ)

Ez a billentyű két feladatot lát el: HOME és CLEAR. Ha ezt a billentyűt leütjük, a kurzor azonnal a képernyő bal felső sarkába ugrik (amit HOME pozíciónak nevezünk). A képernyő többi része változatlan marad. Ha a SHIFT billentyűt lenyomva tartva leütjük a CLEAR/HOME billentyűt, a kurzor a HOME pozícióba ugrik és a képernyő törlődik. A képernyőn csak a bal felső sarokban villogó kurzor marad.

Billentyű	Hatás
CLEAR/HOME	= HOME pozíció
CLEAR/HOME + SHIFT	= Törli a képernyőt



CTRL (KONTROL)

A CTRL billentyű abban hasonlít a SHIFT billentyűhöz, hogy mindig más billentyűvel együtt kell használni. Lenyomva kell tartani, miközben más billentyűket leütünk. A CTRL három esetben használatos:

1. Mint az a COLOUR (szín)billentyűk leírásából kitűnik, a CTRL és egy szín billentyű lenyomása megváltoztatja a képernyőre írt szöveg színét.
2. Képernyőre történő nyomtatást vagy listázást felfüggeszthetünk a CTRL és az S billentyűk lenyomásával.
   (A program újraindításához a RUN/STOP kivételével bármely billentyű leütése elegendő.)
3. A CTRL használható a REVERSE ON/OFF és a FLASH ON/OFF billentyűkkel együtt is.

Ezeket ebben a fejezetben később tárgyaljuk.
Ezen kívül több készen kapható program is használhatja a CTRL billentyűt különleges céljai megvalósítására.

C= (COMMODORE BILLENTYŰ)

A Commodore billentyű nagyon hasonlít a CTRL-hoz, és négy feladat ellátására használható:

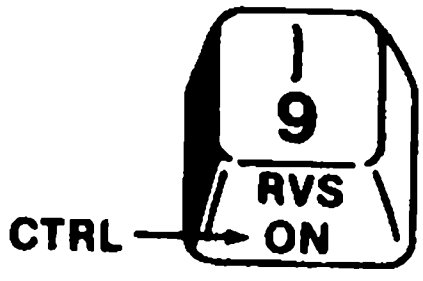
1. A SHIFT billentyűvel együtt használva a C= írógép üzemmódba helyezi a gépet, és lehetővé teszi kisbetűk és nagybetűk használatát.
2. A C= billentyű lehetővé teszi a billentyűk előlapjának bal oldalán található grafikus jelek bevitelét. Csak le kell nyomni a C= billentyűt, és a kívánt grafikus billentyűt.
3. A C= billentyű a CTRL-hoz hasonlóan használható a képernyőre írt szöveg színének megváltoztatására valamelyik szín billentyűvel együtt lenyomva.
4. Ha egy futó listázást akarunk lelassítani (egy sorról sorra megjelenő programot, ami túl gyors ahhoz, hogy követni tudjuk), nyomjuk le a C= billentyűt. A kijelzés futása jelentős mértékben lelassul. Ha elengedjük a billentyűt, visszaáll az eredeti sebesség. (A C= billentyű végre önmagában is csinál valamit!)

RVS ON RVS OFF (INVERZ MÓD BE/KI)

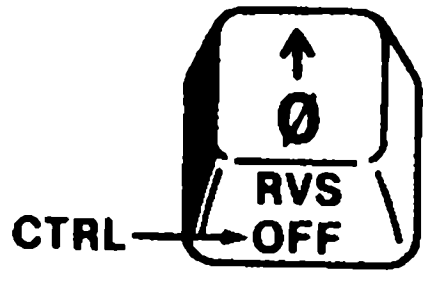
A Commodore 16 lehetővé teszi betűk és számok inverz megjelenítését. Más szóval, ha a kurzor fekete, és a képernyő háttérszíne sárga, a beírt karakter sárgán jelenik meg fekete háttérrel.
A fordított kép eléréséhez mindössze a CTRL és a RVS ON billentyűket kell lenyomni. Minden beírt karakter inverz módban jelenik meg, amíg meg nem

nyomjuk a CTRL és RVS OFF billentyűket, a RETURN-t, vagy az ESC és 0 billentyűket. Így térünk vissza a normál (nem inverz) üzemmódhoz.

Billentyű	+	= Hatás
CTRL	+ RVS ON	= Inverz megjelenítés



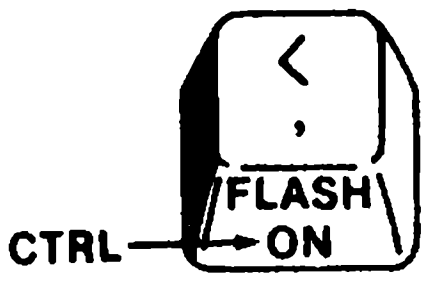
CTRL	+ RVS OFF	= Normál megjelenítés



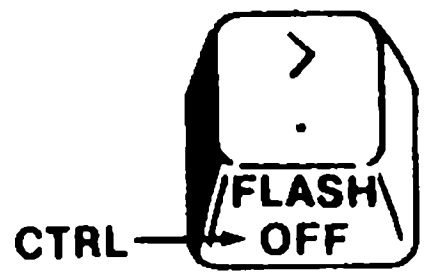
FLASH ON FLASH OFF (VILLOGÁS BE/KI)

A képernyőn megjelenő karaktereket folyamatosan villogtatni lehet a kurzor villogásához ,hasonlóan. A CTRL és a FLASH ON billentyűk lenyomására a beírt szöveg villogni fog. A CTRL és a FLASH OFF billentyűk vagy a RETURN, illetve az ESC és 0 billentyűk lenyomására visszaáll az eredeti (nem villogó) írásmód.

Billentyű	+	= Hatás
CTRL	+ FLASH ON	= A karakterek villognak



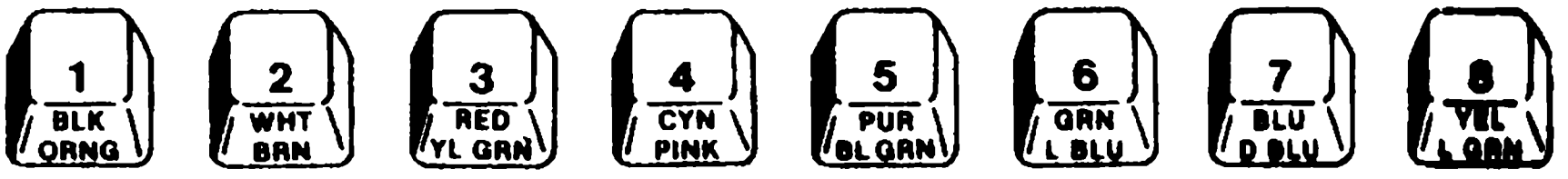
CTRL	+ FLASH OFF	= Normál kijelzés



ESC (ESCAPE)

Az ESC billentyű lehetővé teszi több különleges képernyőszerkesztő feladat elvégzését; beleértve a képernyőablakok létrehozását és kezelését. A képernyőablak a Commodore 16 olyan különleges tulajdonsága, mely lehetővé teszi a képernyő egy részének elkülönített használatát olyan munkaterületként, mely nincs hatással a képernyő többi részére. Az ESC billentyű több ablakszerkesztő funkciót lát el, továbbá több szokásos feladatot, mint például a betoldás, törlés és a képernyő görgetése. Az ESC billentyű összes funkciójára visszatérünk a negyedik fejezetben a képernyőablak tárgyalásakor.

Szín billentyűk



A szín billentyűk, mint láthatjuk, azonosak az 1-től 8-ig terjedő szám billentyűkkel. A CTRL vagy C= billentyűk valamelyikével együtt lenyomva megváltoztatják minden beírt szöveg színét. A Commodore 16 fehér háttérrel és kék kerettel kezd, míg a kurzor feketén villog. A beírt karakterek feketén jelennek meg. A kurzor és a begépelt szöveg színének megváltoztatásához a szín billentyűket kell használni. Az 1-től 8-ig számozott billentyűk előlapjára két szín van felírva. A CTRL billentyűt az előbb említett szám billentyűk valamelyikével lenyomva a felső szint, míg a C= billentyűt egy szám gombbal lenyomva az alul lévő színt kapjuk meg. A szín billentyűk használatával csak a karakterek színét tudjuk megváltoztatni. Ha a háttér vagy a keret színét kívánjuk megváltoztatni, BASIC parancsot kell használnunk. (Részletesen később.)

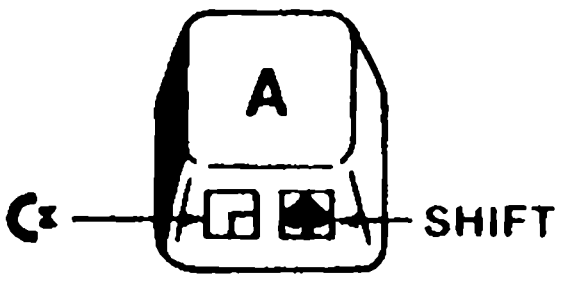
Megjegyzés: A színek változtatásakor ügyelni kell arra, hogy a szám (szín) billentyűt nehogy akár csak a másodperc töredékével is a CTRL illetve C= billentyűk előtt üssük le. Ebben az esetben a szín megváltoztatása helyett csak a szám jelenik meg a képernyőn.

Billentyű	+	= Hatás	Billentyű	+	= Hatás
CTRL	+ 1	= Fekete	C=	+ 1	= Narancs
CTRL	+ 2	= Fehér	C=	+ 2	= Barna
CTRL	+ 3	= Piros	C=	+ 3	= Sárgászöld
CTRL	+ 4	= Türkiz	C=	+ 4	= Rózsaszín
CTRL	+ 5	= Bíbor	C=	+ 5	= Kékeszöld
CTRL	+ 6	= Zöld	C=	+ 6	= Világoskék
CTRL	+ 7	= Kék	C=	+ 7	= Sötétkék
CTRL	+ 8	= Sárga	C=	+ 8	= Világoszöld

Grafikus billentyűk

Minden betű billentyű (és néhány egyéb billentyű) előlapján két különböző jel látható. Ezek a grafikus billentyűk. Bekapcsolás után a Commodore 16 számítógép nagybetűket ír. Ha ezen billentyűk valamelyikét a SHIFT vagy a C= billentyűvel egyszerre nyomjuk le, a billentyű előlapján található gra-

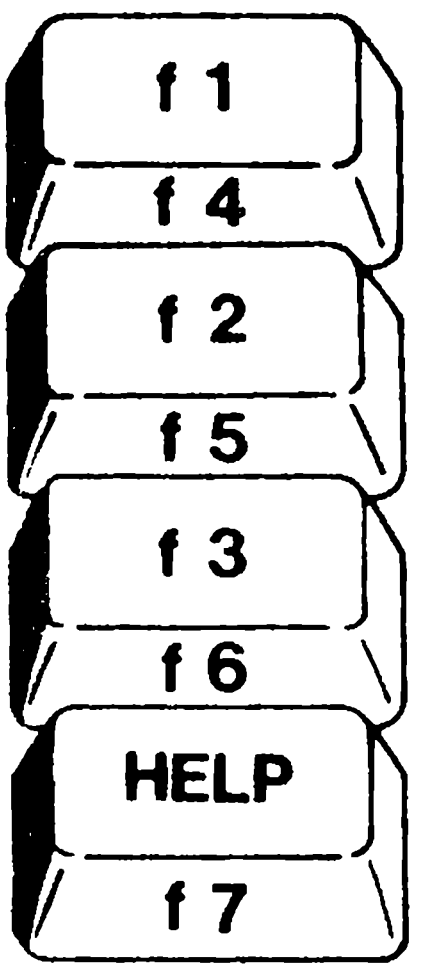
fikus jel kerül a képernyőre. Kiírathatjuk a több mint hatvan grafikus jelből álló karakterkészletet, ahogy az a billentyűk többségének előlapján látható.



Billentyű	+	=	**...tás**
SHIFT	+ A	=	(A billentyű jobb oldalán lévő grafikus jel)
C=	+ A	=	(A billentyű bal oldalán lévő grafikus jel)

Képeket, táblázatokat és alakzatokat hozhatunk létre a grafikus jeleket építőkockákhoz hasonlóan egymás mellé vagy egymás fölé írva. A grafikákat még érdekesebbé tehetjük, ha a szín billentyűk alkalmazásával különböző színekkel "rajzolunk". Próbáljunk néhány grafikus billentyűt használni, hogy lássuk, hogyan működnek! A hatodik fejezet részletesebben tárgyalja a grafikus üzemmódot.
Írógép üzemmódban csak a billentyűk bal oldalán lévő grafikus jeleket használhatjuk (a C= és a megfelelő billentyű lenyomásával). A bal oldali grafikus jelek ideálisak táblázatok, grafikonok és üzleti nyomtatványok létrehozásához.

Funkció billentyűk



A billentyűzet jobb oldalán lévő négy billentyű (elkülönítve a többi billentyűtől) különleges funkció billentyű, melyek időmegtakarítást tesznek lehetővé ismétlődő feladatok végrehajtásakor. A billentyűk tetején f1, f2, f3 és HELP olvasható. Ezeket a funkciókat a billentyűk leütésével valósíthatjuk meg. A billentyűk oldalán f4, f5, f6 és f7 olvasható. Az f1, f2, f3 és HELP SHIFT-tel együtt történő leütésével kaphatjuk meg ezeket a funkciókat.

A billentyűkkel a következő feladatokat lehet ellátni:

Az f1 billentyű valamelyik grafikus üzemmódot indítja, ha kijelöljük a grafikus terület számát és megnyomjuk a RETURN-t. A

GRAPHICS parancs szükséges a grafikus utasítások például CIRCLE vagy PAINT végrehajtásához. A grafikáról a hatodik fejezetben tudhatunk meg többet.

Az **f2** billentyű DLOAD"-ot ír a képernyőre. A DLOAD" beírása helyett csupán a programnevet kell megadni, és a RETURN-t leütni az adott nevű program lemezről történő betöltéséhez.

Az **f3** billentyű a lemezmeghajtóban lévő lemez tartalomjegyzékét listázza ki.

Az **f4** billentyű törli a képernyőt az SCNCLR parancs alkalmazásával.

Az **f5** billentyű DSAVE"-et ír a képernyőre. Csak a megőrzésre szánt program nevét kell beírni és a RETURN-t leütni.

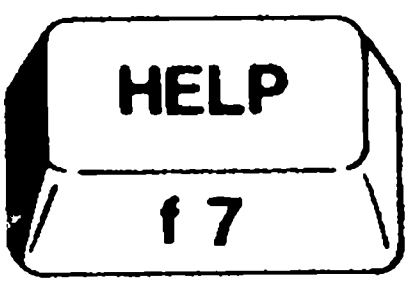
Az **f6** billentyű elindítja a tárban lévő programot.

Az **f7** billentyű az aktuális programot listázza ki.

A **HELP** billentyű a program utasításaiban elkövetett hibákat emeli ki, azok villogó megjelenítésével.

Ezen billentyűk bármelyikét újra definiálhatjuk olyan feladatok elvégzéséhez, melyekre szükségünk van. Az újradefiniálás a KEY parancs alkalmazásával könnyen megvalósítható. Újra definiálhatjuk a billentyűket BASIC programokból, vagy bármikor megváltoztathatjuk őket direkt módban. Egy funkció billentyű újradefiniálására akkor kerülhet sor, ha egy parancs gyakori beírása helyett időt akarunk megtakarítani. Az új definíciók törlődnek a számítógép kikapcsolásával. Annyi billentyűt definiálhatunk újra, amennyit akarunk, és olyan gyakran, ahogyan szükséges.

A HELP BILLENTYŰ (SEGÍTSÉG)



Ha hibát ejtünk egy programban, a számítógép hibaüzenetet ad, jelezve a hiba okát. A hibaüzenetek részletes leírása a könyv végén lévő függelékben található.

Több segítséget kaphatunk a HELP billentyű használatával. A hibaüzenet megjelenése után nyomjuk le a HELP billentyűt a hiba pontos lokalizálásához. A billentyű lenyomása után a hibás sor úgy jelenik meg a képernyőn, hogy a hiba villog. Például:

?SYNTAX ERROR IN LINE 10 . A számítógép ezt írja ki
HELP Megnyomjuk a HELP billentyűt
10 PRONT "COMMODORE A hiba villogva jelenik meg.
COMPUTERS"

3. FEJEZET

Szoftver

- Bevezetés
- Modulok
- Kazetták
- Mágneslemezek

BEVEZETÉS

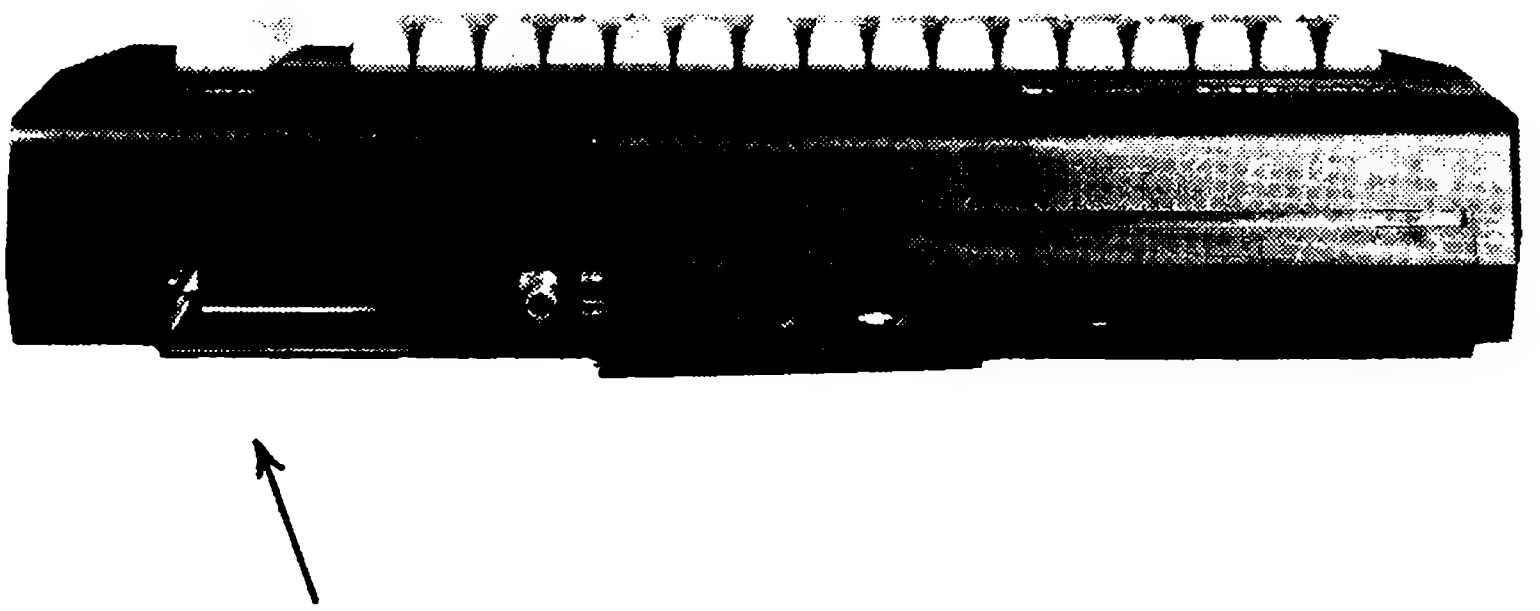
Egy számítógép szoftver nélkül olyan, mint egy pohár, narancsszörp nélkül. Talán hihetetlen, de a szoftvernek köszönhetjük a számítógép használhatóságát, valamint a játékot és szórakozást, amit nyújt. A szoftver a gépbe tölthető és a gépen futtatható programok összessége. A hardver (ebben az esetben a Commodore 16) a szoftvert több formában tudja fogadni: bedugható modulok, szalagra és hajlékony mágneslemezre rögzített programok formájában.
A programokat csak be kell tölteni a Commodore 16 memóriájába. Saját magunk is írhatunk, és tárolhatunk programokat kazettán vagy hajlékony mágneslemezen.

MODULOK

Modulok használata esetén nincs szükség további berendezésre, a teendő csupán annyi, hogy a modult a számítógép hátlapján lévő csatlakozóba dugaszoljuk, és bekapcsoljuk a gépet. A modulok használata esetén az alábbi lépéseket kell követni:

A modulok betöltése

1. lépés. Kapcsoljuk ki a Commodore 16 számítógépet!
FONTOS: A SZÁMÍTÓGÉPET KI KELL KAPCSOLNI, MIELŐTT BEHELYEZZÜK VAGY KIVESSZÜK A MODULT. HA EZT ELMULASZTJUK, KÁROSODHAT A MODUL ÉS A SZÁMÍTÓGÉP IS.
2. lépés. Fordítsuk a modult címkével fölfelé és dugjuk be erősen a számítógép hátoldalán található modulcsatlakozóba (memory expansion felirat)!



A Commodore 16 hátlapja, a modulcsatlakozó nyíllal jelölve

3. lépés. Kapcsoljuk be a Commodore 16-ot!
4. lépés. Kezdjük el a játékot vagy programot a programhoz mellékelt utasítások szerint!

KAZETTÁK

Számos program áll rendelkezésre a Commodore 16-hoz mágnesszalagon. Ezek a kazetták hasonlóak a zenei kazettákhoz, melyeket kazettás magnetofonon vagy sztereo berendezéseken szoktunk lejátszani. A mágnesszalagon tárolt program a modulhoz hasonlóan működik, de rendelkeznünk kell egy további külső készülékkel (úgynevezett perifériával) a kazettának a számítógépbe való betöltéséhez. A kazetták használatához Datassette magnetofonra van szükségünk. Saját programjainkat is tárolhatjuk mágnesszalagon. A következő rész tárgyalja, hogyan kell programokat szalagon megőrizni.
A kazettáról történő programbetöltés lépései ugyanazok, akár előre felvett programot, akár saját magunk által tárolt programot használunk.

MÁGNESSZALAGOK BETÖLTÉSE

1. lépés. Helyezzük be a kazettát a Datassette-be és csukjuk le a fedelét!
2. lépés. Tekerjük vissza a szalagot az elejére a Datassette-en lévő Rewind gomb lenyomásával!
3. lépés. Nyomjuk meg a STOP-ot, ha a szalagot visszatekertük az elejére, írjuk be a LOAD-ot és nyomjuk meg a RETURN-t! A számítógép a következő üzenettel válaszol:

PRESS PLAY ON TAPE (Nyomja meg a play-t a magnón)

4. lépés. Nyomjuk meg a PLAY gombot a Datassetten-en! A képernyő törlődik, amint a kazetta elindul. Ha a gép programot talált, a következő üzenetet küldi a képernyőre:

SEARCHING (keres)
FOUND programnév (a programot megtalálta)

5. lépés. Nyomjuk meg a Commodore billentyűt a megtalált program betöltéséhez! Ha a szalagon egynél több program van, és a Commodore 16 által megtalált program nem az amit szeretnénk, nem kell semmit tennünk. Kis idő után a gép folytatja a keresést.

Ha a programot betöltöttük a READY szó jelenik meg. Ha meg akarjuk állítani a betöltést annak teljes végrehajtása előtt, nyomjuk le a RUN/STOP gombot a billentyűzeten, és a STOP gombot a kazettás egységen. A program betöltése után az elindításhoz írjuk be: **RUN**. A programot ki is listáztathatjuk vagy megváltoztathatjuk, ha az BASIC program.

EGY ADOTT PROGRAM BETÖLTÉSE

Egy szalagon lévő adott program betöltéséhez használjuk a LOAD parancs LOAD "programnév" alakját! A teendő ugyanaz, mint a programnév nélküli LOAD esetén, néhány különbséggel.

1. lépés. Ha a kívánt program neve "ALAP" akkor írjuk be:

LOAD "ALAP"

és nyomjuk meg a RETURN billentyűt!
A számítógép válasza:

PRESS PLAY ON TAPE

2. lépés. Nyomjuk meg a PLAY gombot a kazettás egységen, ez elindítja a keresést. Az ALAP nevű program megtalálása után a következő üzenet jelenik meg:

FOUND ALAP (megtalálta)
LOADING (betölti)

A képernyő törlődik, mialatt a számítógép beolvassa a programot a memóriájába. Ha az egész szalag lemegy FOUND üzenet nélkül, tekerjük vissza a szalagot és próbálkozzunk újra. Ha a számítógép betöltötte a teljes programot, a Commodore 16 kiírja:

READY.

3. lépés. Írjuk be: **RUN**, és nyomjuk meg a RETURN-t! Ekkor a Commodore 16 elindítja (végrehajtja) az ALAP nevű programot.

PROGRAMOK FELVÉTELE KAZETTÁRA

Ha magunk írunk programot és meg akarjuk őrizni későbbi használat vagy módosítás céljából, tárolhatjuk a programot (SAVE). Ha egy programot megőrzünk, olyan formában vesszük fel szalagra vagy mágneslemezre, ami lehetővé teszi a program visszatöltését futtatás vagy változtatás céljából.

Ha egy programot kazettán akarunk megőrizni, kövessük az alábbi lépéseket:

1. lépés. Írjuk be:

SAVE"programnév"

Az általunk használt programnév bármi lehet, de nem lehet hosszabb 16 karakternél.

2. lépés. Nyomjuk le a RETURN billentyűt! A számítógép a következőt üzeni:

PRESS RECORD AND PLAY ON TAPE

(Nyomja le a record és play gombokat a magnón)

3. lépés. Nyomjuk le a RECORD és PLAY gombokat a kazettás egységen! A képernyő törlődik. Ha a program rögzítése befejeződött, a READY szó jelenik meg a képernyőn.

Példák SAVE parancsokra kazettás egység esetén:

SAVE "A NAP"	Ezekkel a nevekkel fogjuk
SAVE "SAJÁT MAGA"	megőrizni a programokat

MEGJEGYZÉS: Ha kazettára veszünk fel egy programot, mindig ügyeljünk a szalag pozicionálására.

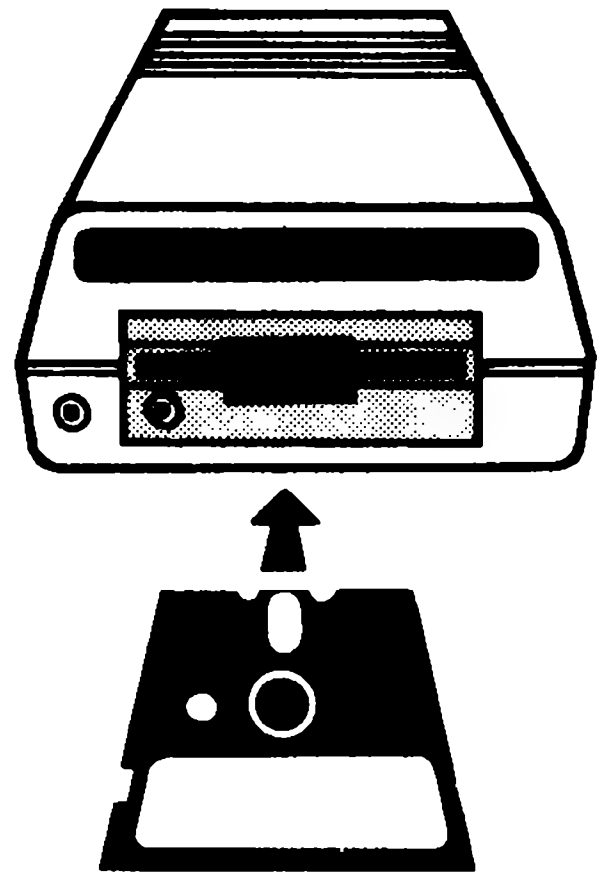
HAJLÉKONY MÁGNESLEMEZEK

A hajlékony mágneslemezek gyorsak és könnyen használhatók. A mágneslemezeket és a meghajtóegységet gondosan kell kezelni. A hajlékony mágneslemezeket nevezik diszknek, diszkettnek, floppy diszknek vagy floppynak; mind ugyanazt jelentik. A kazettáktól eltérően a lemezeket csak be kell helyezni a meghajtóegységbe és be kell írni a betöltésre vagy megőrzésre vonatkozó parancsot, nem kell gombokat megnyomni az indításhoz. A lemezmeghajtó elején kis jelzőfények találhatók. A zöld fény a tápfeszültség bekapcsolását jelzi. A piros fény két dolgot jelenthet. Használat közben, mikor a program betöltése vagy mentése folyik, a lemez forgása közben világít. Ha probléma van a lemezzel vagy a meghajtóval, a piros lámpa villog a lemez leállása után is.

PROGRAMOK BETÖLTÉSE MÁGNESLEMEZRŐL

1. lépés. Győződjünk meg róla, hogy a készülék a hálózathoz és a soros kábellel a géphez csatlakoztatva van! Azután kapcsoljuk be a meghajtó egységet!

2. lépés. Helyezzük be a lemezt a meghajtó egységbe! A lemez címkézett fele felfelé nézzen. Úgy helyezzük be a lemezt a nyílásba, hogy a címkézett rész kerüljön be utoljára. A lemez oldalán egy kis bevágás található (lehet, hogy ragasztószalag fedi). Ennek a bevágásnak bal oldalon kell lennie a lemez behelyezése során, ha a meghajtóegységgel szemben vagyunk. Ütközésig toljuk be a lemezt!



3. lépés. Zárjuk be a meghajtóegység védőajtaját, miután a lemezt behelyeztük!
4. lépés. Írjuk be:

DLOAD "programnév" (a betöltendő program neve)

(Időmegtakarítás céljából megnyomhatjuk a 2. funkció billentyűt és beírhatjuk a program nevét, majd a második idézőjelet.)
5. lépés. Nyomjuk le a RETURN billentyűt! A lemez forogni kezd, és a képernyőn megjelenik:

SEARCHING FOR "programnév" (Keresi)
FOUND "programnév" (Megtalálta)
LOADING (Betölti)
READY.

6. lépés. A program kész az indításra. Írjuk be: **RUN** és nyomjuk meg a RETURN-t a program elindításához!
A meghajtóegységen lévő piros fény a DLOAD befejezése után is villog hiba esetén. Írjuk be:

? DS$ és nyomjuk le a RETURN-t a hiba kiderítéséhez.

Példák DLOAD parancsokra:

DLOAD "*" Betölti a diszken lévő első programot.
DLOAD "ADAT" Betölti az Adat nevű programot.
DLOAD "SOF*" Betölti a diszken lévő első olyan programot, melynek neve SOF-fal kezdődik.

LEMEZ INICIALIZÁLÁSA

Az inicializálás új, üres lemezt készít elő a használatra. Az üres diszkeket használat előtt inicializálni kell a HEADER paranccsal.

FONTOS: Sose inicializáljunk olyan diszket, amely nem üres, hacsak nem akarjuk az egész lemezt törölni. Az inicializálás mindent letöröl a lemezről.

A HEADER parancs formája a következő:

HEADER "Lemeznév" [,U meghajtó címe] [,I azonosító], D meghajtó száma

- A név az egész lemez neve. A lemeznek bármilyen nevet adhatunk, de nem lehet hosszabb 16 karakternél.
- A meghajtó címe jelöli ki a periféria címét, mely alapértelmezés szerint 8 (első meghajtó).
- Az azonosító a nagy I betű és két betű vagy szám, mint például I21, IR5 stb. A lemezeknek tetszés szerinti azonosítót választhatunk, de célszerű minden lemeznek különböző azonosítót adni az összetévesztés elkerülése érdekében.
  (Megjegyzés: az azonosító első karaktere nem lehet F.)
- Ha kettős meghajtóegységünk van (két lemez meghajtására alkalmas berendezés), akkor a 0 vagy 1 a meghajtó számát jelöli. A meghajtó számának MINDIG szerepelnie kell a parancsban, még egyes meghajtó esetén is (D0).

BIZTOS BENNE?

Amint megnyomjuk a RETURN gombot a HEADER parancs beírása után, a Commodore 16 megkérdezi:

ARE YOU SURE? (Biztos benne?)

Ez az utolsó lehetőség, hogy elhatározásunkat megváltoztassuk. Ha inicializálni akarjuk a lemezt, írjuk be:

YES vagy **Y** és nyomjuk meg a RETURN-t.

Ha úgy döntünk, hogy nem inicializáljuk a lemezt, írjuk be:

NO vagy **N** és nyomjuk meg a RETURN-t.

Álljon itt néhány példa a HEADER parancsra:

HEADER "POSTA", U8, I07, D0
HEADER "ANYAG", U8, IS2, D0

MEGJEGYZÉS: Ha minden állományt törölni akar egy lemezen (szemben egy új lemez inicializálásával), azonosító nélkül adja ki a HEADER parancsot. Ebben az esetben csak a lemezen lévő állományok fognak törlődni.

Most, miután tudjuk, hogyan inicializáljunk lemezeket, készen állunk a Commodore 16-on (vagy bármely más Commodore számítógépen) programok lemezre való mentésére.
A könyv végén található BASIC ismerettár több információt tartalmaz a HEADER parancsról.

PROGRAMOK MEGŐRZÉSE DISZKETTEN

Ha egy megírt programot újra használni akarunk, ki kell mentenünk, mielőtt egy másik programot betöltünk vagy a számítógépet kikapcsoljuk. Ha ezt elmulasztjuk, a program elvész. Ha egy felvett programot módosítunk, újra ki kell mentenünk, ha meg akarjuk őrizni az új változatot.
Ha egy programot újra kimentünk, a régi változatot felülírjuk az újjal. Ha mind a régi, mind az új változatot meg akarjuk tartani, akkor az új változatnak felvételkor más nevet kell adnunk.
Program lemezre viteléhez a következő lépéseket kövessük:
1. lépés. Írjuk be.

DSAVE "programnév"

2. lépés. Nyomjuk meg a RETURN-t. A számítógép a következő üzenetet írja ki:

SAVING 0 "programnév"
READY

Például:

DSAVE "PROGRAM5" A programnév maximum 16 karakter hosszú lehet

Ha a lemezmeghajtón lévő piros lámpa a DSAVE befejezése után is villog, valami hiba történt. Írjuk be a

?DS$ parancsot (és nyomjuk meg a RETURN-t)

a hiba azonosításához.
Ha egy írás ellen védett lemezre kíséreltünk meg írást, a meghajtót **előbb KI**, majd újra BE kell kapcsolni.

A DIRECTORY (KATALÓGUS) PARANCS

Ha programot viszünk fel lemezre, a számítógép a lemezen lévő **összes** állományról listát készít. Ha kíváncsiak vagyunk a lemez tartalmára, a listát mint egy tartalomjegyzéket a DIRECTORY paranccsal kiírathatjuk.
Írjuk be:

DIRECTORY majd nyomjuk meg a RETURN-t
(vagy nyomjuk meg a 3-as funkció billentyűt)!

Amint a RETURN-t megnyomjuk, a Commodore 16 mindent kiír, ami a tartalomjegyzékben van.
Kiirathatjuk a tartalomjegyzék egy részét is:

DIRECTORY "MY*"	A lemezen lévő minden olyan állományt kilistáz, amelynek neve MY betűkkel kezdődik.
DIRECTORY "*= PRG"	Minden programot kilistáz.
DIRECTORY "*= SEQ"	Minden soros állományt kilistáz.

4. FEJEZET

Az első lépések

- Bevezetés
- A képernyő

Inverz kijelzés és színváltoztatás

- Az első lépés
- Hibajavítás
- Képernyő ablakok

BEVEZETÉS

A fejezet célja, hogy elkezdjük a Commodore 16 jellemzőivel és képességével való megismerkedést, és megtegyük a számítógép programozásának első lépéseit.

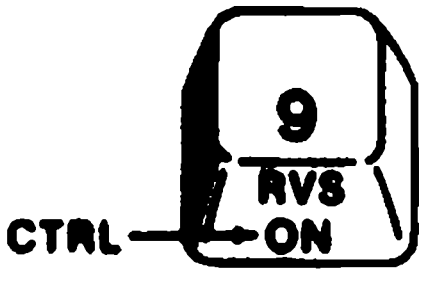
A képernyő

A Commodore 16 működése nyomon követhető a TV vagy monitor képernyőjén. Minden, amit beírunk, amit a számítógép mondani akar, és minden betöltött program neve megjelenik a képernyőn, legyen az színes, vagy fekete-fehér TV, vagy a Commodore színes monitora. A Commodore 16 képernyője függetlenül a használt TV vagy monitor típusától 40 oszlop széles. Ez azt jelenti, hogy a képernyő bal szélétől a jobb széléig negyven karaktert tudunk írni. A képernyőn 25 sor van, így egy teljesen kitöltött képernyő 1000 (25 x 40) karaktert tartalmazhat. A Commodore 16 képernyője három elemből áll: a karakterekből (betűk, számok és grafikus jelek), a háttérből és a keretből. Bekapcsolás után a karakterek feketék, a háttér fehér, és a keret világoskék. A második fejezetben már láttuk, hogyan lehet változtatni a karakterek színét a szám-, valamint a CTRL vagy C= billentyűk együttes használatával. Ilymódon a karaktereknek 16 különböző színt adhatunk. Van még néhány dolog, amivel a karakterek kiírását érdekesebbé tehetjük.

Inverz kijelzés és színváltoztatás

A képernyőn nemcsak a kurzor és a beírt (valamint a számítógép által kijelzett) karakterek színét tudjuk megváltoztatni, hanem föl tudjuk cserélni a karakterek és a háttér színét is. Ez azt jelenti, hogy ha a karakterek fehér háttérben feketék, és bekapcsoljuk az inverz kijelzést (CTRL és 9 beírásával), minden ami az ernyőn megjelenik, megfordul: a karakterek fehérek lesznek, míg a karakterek mögötti háttér fekete. Kísérletezhetünk a karakterszínek változtatásával és az inverz kijelzés használatával az alábbi utasítások szerint:

1.lépés. Tartsuk lenyomva a CTRL billentyűt és nyomjuk meg az RVS ON billentyűt!



2. lépés. Engedjük el a billentyűket, és tartsuk lenyomva a szóköz billentyűt!
3. lépés. Tartsuk lenyomva a szóköz billentyűt ameddig csak akarjuk! Mialatt lenyomva tartjuk a szóközt, a képernyőn a betűkkel azonos színű sor halad tovább. Ha a csík a sor végére ér, a következő sorban folytatódik.
4. lépés. Engedjük el a szóközt (de ne nyomjuk meg a RETURN-t)!
5. lépés. Tartsuk lenyomva a CTRL billentyűt és nyomjuk meg a szín billentyűk egyikét (de ne olyan színt, amilyen már van a képernyőn). A kurzor rögtön olyan színűre változik, amilyen billentyűt lenyomtunk.
6. lépés. Tartsuk újra lenyomva a szóköz billentyűt! A számítógép most más színű csíkot rajzol. Folytassuk a színek változtatását a CTRL vagy C= billentyűkkel, illetve a szín billentyűkkel. Tartsuk újra lenyomva a szóközt különböző színű csíkok húzásához!
7. lépés. Kapcsoljuk ki az inverz kijelzést a CTRL és az RVS OFF billentyűk megnyomásával. A RETURN billentyű szintén megszünteti az inverz kijelzést.

RÖVID KÖZBEIKTATÁS: Betűket, számokat és írásjeleket tudunk inverz kijelzéssel megjeleníteni. Az inverz betűk elegáns fejlécet alkotnak. Arra is használhatjuk őket, hogy fontos szavakat és számokat kihangsúlyozzunk. Grafikus jeleket is írhatunk fordítottan hasonló hatással.
Versenypálya létrehozásához válasszunk ki egy tetszés szerinti színt, és írjuk be a – jelet (a SHIFT és E megnyomásával) egy teljes soron keresztül! Ezután térjünk át inverz kijelzésre: töltsük ki a következő sort ugyanezzel a jellel, csak fordítottan! A következő sorba írjunk SHIFT-et és R-t (még inverz kijelzéssel)! Ezután kapcsoljuk ki az inverz kijelölést, és töltsük ki az utolsó sort SHIFT-tel és R-rel (már nem inverz). A képernyőn létrejött a versenypálya a választott színben; talán szerény eredmény, de kölcsönöz a képernyőnek bizonyos versenypálya nézetet...

AZ ELSŐ LÉPÉS

Mi is az a program? Egy gyakorlott programozó azt mondhatná, hogy a program rendezett utasítások sorozata, mely a számítógépet sorrendi lépések megtételére utasítja egy célszerű eredmény elérése érdekében. Ebből annyit tanulhatunk, hogy gyakorlott programozótól nem érdemes megkérdezni, hogy mi az a program.
A következő, bár csak két sor hosszúságú, programnak minősül. Pontosan úgy írjuk be, ahogy itt szerepel. Ne hagyjuk ki a számokat a sorok elején, mivel ezek sorszámok. Ezek mondják meg a számítógépnek, hogy milyen sorrendben kell értelmezni és végrehajtani a program sorait. A sorok végén ne mulasszuk el lenyomni a RETURN billentyűt!

10 PRINT "1. PROGRAM" Ez a sor arra utasítja a számítógépet, hogy írja ki a képernyőre az 1. PROGRAM szavakat.

20 GOTO 10 Ez a sor arra utasítja a számítógépet, hogy menjen vissza a 10-es sorra és írja ki újra az 1. PROGRAM szöveget.

RUN Amikor ezt beírjuk és lenyomjuk a RETURN billentyűt, a számítógép végrehajtja azt, amire a fenti sorok utasítják.

Nyomjuk meg a RUN/STOP billentyűt a program megállításához!

Miért írja ki a számítógép az 1. PROGRAM szöveget olyan sokszor? A PRINT olyan parancs, ami a számítógépet arra utasítja, hogy az idézőjelek közti szöveget írja ki a képernyőre. Mikor a számítógép ezt a sort elolvassa, végrehajtja a parancsot. A GOTO arra utasítja a számítógépet, hogy menjen vissza a 10-es sorra és hajtsa végre az ott lévő parancsot (PRINT "1. PROGRAM"). Minden alkalommal, amikor a számítógép a 20-as sorra megy végrehajtja a GOTO parancsot, és újból és újból visszatér a 10-es sorra. Ezt az ismétlődést huroknak nevezzük. A RUN/STOP billentyű beavatkozik a számítógépen futó bármely programba, és visszaadja a vezérlést a billentyűzetnek.

ÖTLET: Ha le akarjuk lassítani ezt a programot (vagy bármely másikat) a program megállítása nélkül, tartsuk lenyomva a C= billentyűt.

Írjuk be a következőt, minden beírás után megnyomva a RETURN billentyűt:

NEW Arra utasítja a számítógépet, hogy felejtse el az utolsó programot, és készüljön fel egy új fogadására.

A számítógép válasza:

READY Ezt nem kell beírni; ezzel tudatja a számítógép, hogy készen áll új parancsok vagy programok fogadására.

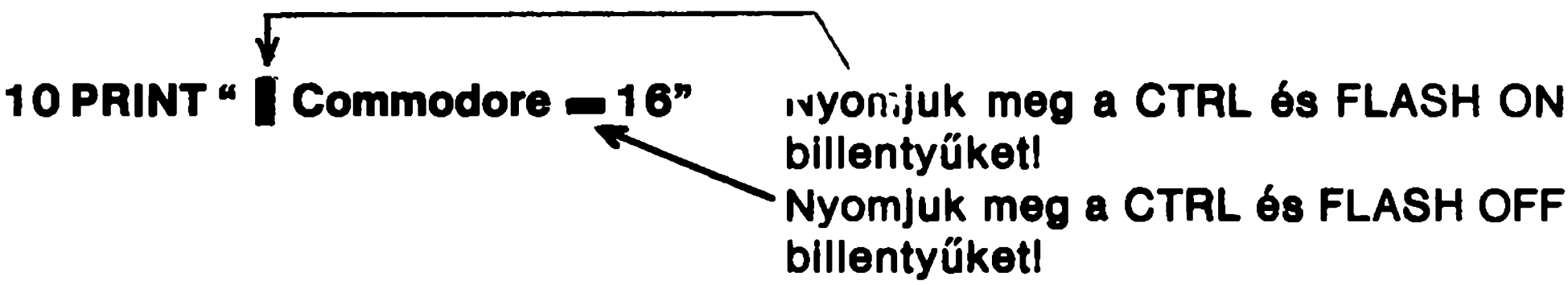
10 PRINT "R COMMODORE – 16"

Nyomjuk meg a CTRL és RVS ON billentyűket!

Nyomjuk meg a CTRL és RVS OFF billentyűket!

Figyeljük meg, mi történik, ha beírjuk a REV ON és REV OFF jeleket az idézőjeleken belülre! Ezek az ernyőn inverz karakterként jelennek meg. Írjunk be RUN-t és nyomjuk meg a RETURN-t! Amikor a program fut, a Commodore 16 értelmezi ezeket az inverz karaktereket, és az utasításnak megfelelően inverz nyomtatást hajt végre. Ugyanezek az elvek érvényesek (inverz karakter jelenik meg az ernyőn beírásunk szemléltetésére) a PRINT utasításban szereplő egyéb műveletek, például a kurzor mozgatása vagy a színek változtatása esetén is, a program futtatása közben.

Próbáljuk ki ugyanezt a sort, de a RVS ON-t és RVS OFF-ot FLASH ON-nal és FLASH OFF-fal (a < és > billentyűk) helyettesítve.



Ha ezt az utasítást végrehajtjuk, csak egyszer írja ki a Commodore 16 szöveget, de a Commodore szó folyamatosan villog még akkor is, ha a program már lefutott. Villogó nyomtatás használata folyamatos villogást eredményez. Mindkét egysoros program azt szemlélteti, hogyan kell inverz kijelzést vagy villogtatást egy programsorban végrehajtani. Ily módon más utasításokat is végrehajthatunk (pl. színváltoztatást, stb.).

PARANCSOK VÉGREHAJTÁSA

Feltűnhetett, hogy néhány utasítást egyetlen szóként írtunk be (például NEW), míg más sorokban sorszámot kellett írnunk, melyet utasítások követtek. Ez azért van, mert két üzemmód létezik, amelyeken párbeszédet folytatunk a számítógéppel. Mindkettő egy kifejezésekből álló nyelvre épül, melyet a számítógép megért. A legcélravezetőbb (és a Commodore 16-ba beépített) nyelv a BASIC.
A Commodore 16 a Commodore BASIC 3.5 nevű változatot érti. A BASIC kifejezések (vagy kulcsszavak) alkotják mindkét üzemmód lelkét. Az előző üzemmód, a KÖZVETLEN MÓD a számítógépet a BASIC parancsok azonnali végrehajtására utasítja. A számítógépbe beírt parancsot a RETURN billentyű megnyomásával hajtjuk végre. Ezt az üzemmódot DIREKT MÓD-nak is nevezzük. A másik üzemmód PROGRAM- vagy INDIREKT MÓD-ként ismert. Program módban a program sorai sorszámokat, és BASIC utasításokat tartalmaznak. A program végrehajtására akkor kerül sor, amikor a RUN parancsot beírjuk. Ilyenkor az utasítások végrehajtása a sorszámok sorrendjében történik (növekvő sorszám szerint). Számítógép-program írásához nyilvánvalóan a program módot alkalmazzuk.

HIBAJAVÍTÁS

A hibák a programozás természetes vélejárói. A programírás közben elkövetett hibákat meg kell keresni és ki kell javítani, hogy a program helyesen fusson. Erre szolgál a HELP billentyű. A hibák leggyakoribb fajtája – a gépelési hiba – teljesen tönkre tud tenni egy programot. Programírás közben nemcsak gépelési gyakorlatot szerzünk, hanem a gépelési hibák keresésében és javításában is jártasak leszünk. A gépelési hibák kijavítására több lehetőségünk van.

Ne feledkezzünk meg arról, hogy bármely változtatás végrehajtásához, akár a számítógép által végrehajtandó, akár a memóriába írandó változtatásról van szó, meg kell nyomnunk a RETURN billentyűt!

1. Sor javítása felülírással
A KURZOR BILLENTYŰK segítségével lépjünk a sorban a hiba helyére! Így egyszerűen felülírhatjuk, amit változtatni akarunk. Nyomjuk meg a RETURN-t a javítás elvégzése után!
Példa:

10 PRINT "EGY ÓRA VAN"

Ha az időt két órára akarjuk változtatni, vigyük a kurzort az egy szóban lévő e-re.

10 PRINT "EGY ÓRA VAN"

Írjunk KETTŐ-t az EGY fölé és nyomjuk meg a RETURN-t!

10 PRINT "KÉT ÓRA VAN"

MEGJEGYZÉS: Ha számozott programsorokkal dolgozunk, nem kell a sor végén lennünk ahhoz, hogy megnyomjuk a RETURN billentyűt. A Commodore 16 az egész sorra emlékszik akkor is, ha a RETURN billentyűt a sor közepén nyomjuk meg.

2. Üres helyek hozhatók létre egy szóban vagy sorban az INST billentyű segítségével (SHIFT és INST/DEL billentyűk együttes lenyomásával). Addig tartsuk lenyomva a billentyűt, míg elegendő számú üres hely keletkezik. (Jegyezzük meg, hogy a kurzor helyben marad és az üres helyek a jobb oldalon nyílnak.) Ezután beírhatjuk, amit be akarunk toldani.

10 PRINT "CORE"

Ha ezt Commodorra kívánjuk változtatni, mozgassuk a kurzort az O-ra és tartsuk lenyomva a SHIFT és INST billentyűket, míg elegendő hely keletkezik. Ne törődjünk a szóközök számolásával! Találomra hagyjunk ki helyet, és legföljebb újra betoldunk, ha nem lesz elegendő!

10 PRINT "C ORE"

Most beírhatjuk a többi betűt:

10 PRINT "COMMODORE"

3. Karakterek és szóközök törlése a DEL billentyűvel lehetséges (az INST/DEL billentyű önmagában). Ez a billentyű közvetlenül a kurzor bal oldalán álló karaktereket vagy szóközöket törli.

10 PRINT"FÉLÉVES TERV"

Ezt HAVI TERV-re változtatjuk, ha a kurzort A FÉLÉVES második É betűjére visszük, az INST/DEL billentyűt háromszor megnyomjuk, és beírjuk, hogy **HAVI:**

10 PRINT"ÉVES TERV" (Az INST/DEL-ek után)

10 PRINT "HAVI TERV" (A HAVI begépelése után)

ezután nyomjuk meg a RETURN billentyűt.

4. Egy sort bármikor újraírhatunk, még a program futtatása után is. A Commodore 16 automatikusan helyettesíti a régi sort az újjal, amint megnyomjuk a RETURN billentyűt. A régi sor még látható az ernyőn, de a számítógép figyelmen kívül hagyja. Ha két utasításnak ugyanaz a sorszáma, a Commodore 16 csak a később beírtat veszi figyelembe.
Például ha egy rövid programban a COLOR parancsot használjuk a képernyő háttérszínének megváltoztatására, és hibát követtünk el:

10 COKOR 0,3 Hiba
20 PRINT "COMMODORE 16"

Nyomjuk meg a RETURN-t, és írjuk újra a 10-es sort helyesen!

10 COLOR 0,3

Az első 10-es sort a második 10-es sorszámú sor fölülírja. Az eredmény a LIST paranccsal ellenőrizhető. Ennek hatására a számítógép memóriájában tárolt program sorról sorra megjelenik a képernyőn. A sorok a sorszámok sorrendjében jelennek meg, a felülírt sorok pedig nem szerepelnek a listán.

LIST RETURN

A képernyőn ez jelenik meg:

10 COLOR 0,3
20 PRINT "COMMODORE 16"

Megfigyelhetjük a COLOR parancs hatását a példában. Teljesebb magyarázatot a hatodik fejezetben adunk, vagy nézze meg a COLOR parancsot a BASIC ismerettárban.
Sorok helyettesítése egy programban jó módja a kísérletezésnek. Egy sor helyettesítésekor az új sornak semmiben sem kell hasonlítania a régire.
Például a COLOR javítása helyett beírhatjuk a következő sort:

10 PRINT "COMMODORE 4-SZER COMMODORE 4 = " Szóköz

Indítsuk el a programot, és nézzük meg, mi történik!

5. Nem kívánt sort egyszerűen törölhetünk a sorszám beírásával és a RETURN megnyomásával. A számítógép figyelmen kívül hagyja a sort még akkor is, ha az még a képernyőn szerepel. Írjuk be a LIST parancsot a program listázása céljából, hogy megbizonyosodjunk arról, hogy a sor eltűnt a programból.

```
10 PRINT "COMMODORE 4-SZER COMMODORE 4 = "
20 PRINT "COMMODORE 16"
```

Írjunk be:

10-et és RETURNT-t!
LIST

Az eredmény a következő:

```
20 PRINT "COMMODORE 16"
```

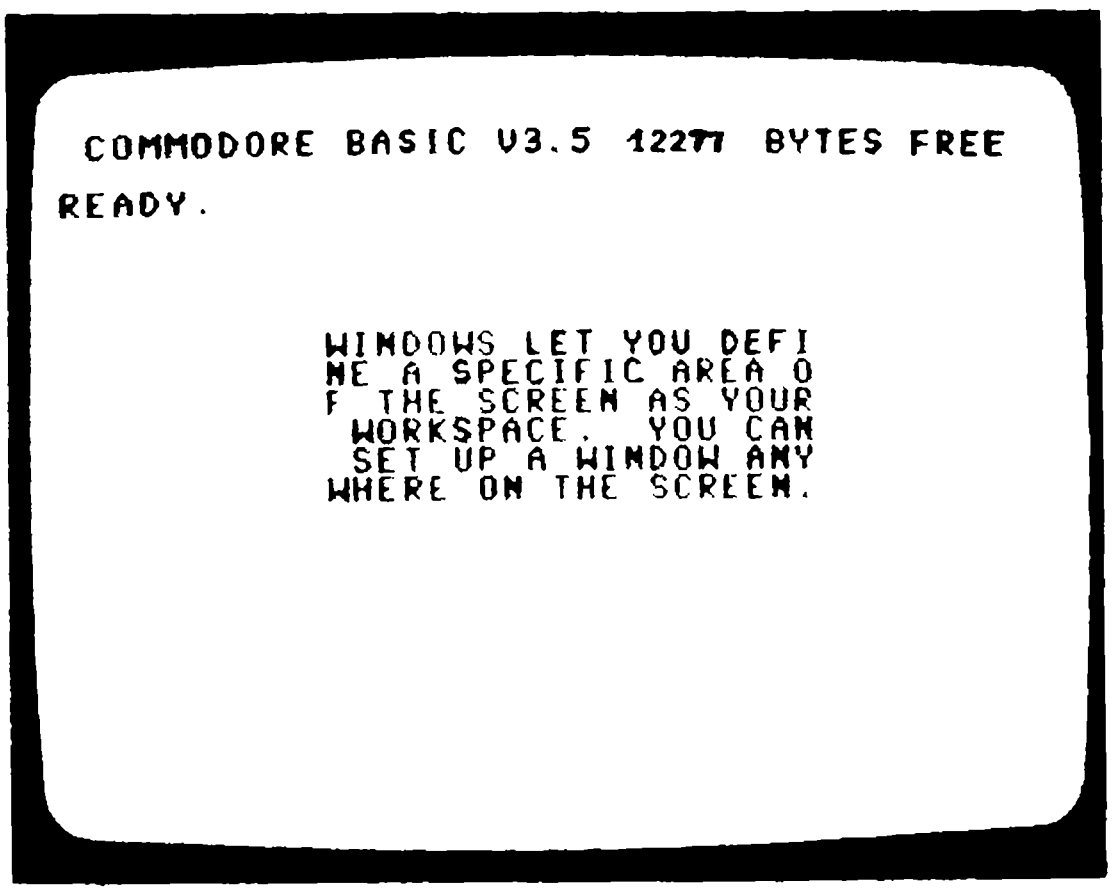
A KÉPERNYŐ TÖRLÉSE

Előfordulhat, hogy a képernyő túlzsúfolt, vagy újra akarjuk szervezni a kiírt szöveget, vagy egyszerűen nem tetszik, ami a képernyőn van, és meg akarunk szabadulni tőle. Többféleképpen tehetjük ezt meg, érintetlenül hagyva a számítógép memóriájában lévő programot, vagy az ernyővel együtt a memóriát is törölve.
A képernyő törlésének egyik módja, hogy a SPACE billentyűt folyamatosan lenyomva tartjuk mindaddig, amíg a képernyőn lévő 1000 karaktert törli (ez eltart néhány percig). Ha a Felhasználói Kézikönyvet eddig elolvasta, elképzelhető, hogy van türelme a képernyőt ilymódon törölni, de az is, hogy intelligenciájával rájött, hogy kell lennie ennél jobb módszernek is. Íme néhány jobb megoldás:

1. Tartsuk lenyomva a ↓ billentyűt mindaddig, míg a képernyő fölfelé görgetésével eltűnik a képernyő tartalma.
2. Nyomjuk meg a CLR/HOME és SHIFT gombokat együttesen. Ez törli a képernyőt, és a kurzort a bal felső sarokba viszi (HOME pozíció).
3. Írjuk be:
   SCNCLR és nyomjuk meg a RETURN billentyűt!
   Ez egy BASIC nyelvű parancs, amit végrehajtva a számítógép törli a képernyőt.
4. Nyomjuk meg a RESET gombot! Ez alaphelyzetbe hozza a gépet, visszaállítja a kezdő képernyőt és törli a memóriában lévő programokat.

Ha csak a memóriát akarjuk törölni, de a képernyőt nem, írjuk be a NEW parancsot és nyomjuk meg a RETURN-t.
Ez egy BASIC nyelvű parancs, ami arra utasítja a számítógépet, hogy törölje a memóriáját. A 16 a COMMODORE 16-ban a számítógépben lévő tároló méretére (16K) utal, amely programok tárolására szolgál és memóriaként ismert. Ha a képernyőt az első három módszer valamelyikével töröljük, csak a képernyőn lévő karaktereket töröljük, de a számítógép memóriája változatlan marad. A két utolsó módszer törli a számítógép memóriáját is, így újra a teljes memóriakapacitás áll rendelkezésre új programok beírására.

Képernyő ablakok



Az ablakok lehetővé teszik , hogy munkaterületként kijelöljük a képernyő egy részét. Minden amit beírunk (begépelt sorok, programlisták, stb.), az ablak beállítása után az ablak keretein belül jelenik meg és nem befolyásolja az ablakon kívüli részeket. A képernyőn bárhol kijelölhetünk egy ablakot. Ablak kijelöléséhez kövessük az alábbi lépéseket:

1. Mozgassuk a kurzort a képernyő azon pontjára, amely az ablak bal felső sarka lesz!
2. Nyomjuk meg az ESC billentyűt, utána a T betűt!
3. Mozgassuk a kurzort arra a helyre, ahol az ablak jobb alsó sarka lesz!
4. Nyomjuk meg az ESC billentyűt, majd a B-t! Ezzel létrehoztuk az ablakot.

Minden kimenet a definiált ablakra korlátozódik. Az ablak törléséhez nyomjuk meg kétszer a CLEAR/HOME billentyűt. Az ablak ezzel törlődik, és a kurzor a bal felső sarokba kerül.
Az ablakot és az ablakon belül lévő szöveget az ESC gomb használatával kezelhetjük. A szerkesztés, betoldás és törlés, listázás és az ablak méretének változtatása az ESC és egy azt követő valamely billentyű megnyomásával hajtható végre. Az adott művelet végrehajtásához nyomjuk meg a megfelelő billentyűt az ESC megnyomása után!

BILLENTYŰ	FUNKCIÓ
A	Automatikus betoldás
B	A képernyőablak jobb alsó sarkát határozza meg (a kurzor aktuális helyén)
C	Törli az automatikus betoldást
D	Törli az aktuális sort
I	Betold egy sort
J	Az aktuális sor kezdetére lép
K	Az aktuális sor végére lép
L	Görgetés engedélyezése
M	Görgetés tiltása
N	Visszatérés a normál képernyőmérethez
O	Törli a betoldást, idézőjelet, inverz és villogó üzemmódokat
P	A sor elejétől mindent töröl a kurzorig
Q	Mindent töröl a kurzortól a sor végéig
R	Csökkenti a képernyő méretét
T	Beállítja a képernyőablak bal felső sarkát
V	Képernyő görgetése fölfelé
W	Képernyő görgetése lefelé
X	Törli az előző ESC funkciót

5. FEJEZET

Számok és számítások

- Bevezetés
- Számok és műveletek
- Számítások
- A képernyőre történő nyomtatásról részletesebben
- Változók
- Numerikus függvények
- A felhasználó által definiált függvények

BEVEZETÉS

Nem kell matematikai zseninek lennünk ahhoz, hogy megértsük és használni tudjuk a Commodore 16 matematikai képességeit. Az alapműveletek, az összeadás, kivonás, szorzás és osztás elvégzésén kívül a számítógépet arra is használhatjuk, hogy bonyolultabb függvényeket, például négyzetgyökvonást és szinuszt is kiszámoljon. Fogunk még hallani különböző típusú változókról és azok használatáról. Számítógépünk egyforma hatékonysággal kezeli akár az egyjegyű egész számokat, akár a 10^{38} nagyságrendű számokat. A számításokat közvetlenül vagy programba írva hajthatjuk végre. Végül a fejezet röviden ismerteti, hogyan definiálhatunk magunk is függvényeket.

SZÁMOK ÉS MŰVELETEK

Számítógépünket egyszerű kalkulátorként is használhatjuk. A szokásos plusz és mínusz műveleti jelek mellett a Commodore 16 a * (csillag) jelet használja a szorzás, és a / (per) jelet az osztás és a törtek jelölésére. (A számítógépek általában a csillag jelet használják az x jel helyett a szorzás jelölésére, mivel a számítógép nem tudná megkülönböztetni az x betűt és az x matematikai szimbólumot.) A műveleti jeleket és számokat közvetlen üzemmódban (sorszámok nélkül), vagy utasításokban egyaránt használhatjuk.
Ha a matematikai műveleteket közvetlenül akarjuk végrehajtani, a PRINT utasításba IDÉZŐJELEK NÉLKÜL írjuk be a számokat és műveleti jeleket. Ha a számok és műveleti jelek idézőjelben vannak (például PRINT "2+ 2") a számítógép pontosan azt fogja kiírni, ami az idézőjelben van, ahelyett hogy az összeadást elvégezné. A számítógéppel a 2+ 2 összeadást a következőképpen végeztethetjük el:

PRINT 2+ 2

BASIC matematikai műveleti jelek		BASIC relációk műveleti jelei	
Összeadás	+	Nagyobb	>
Kivonás	–	Kisebb	<
Osztás és tört	/	Egyenlő	=
Szorzás	*	Nagyobb vagy egyenlő	=> vagy >=
Hatványozás (SHIFT és 0)	↑	Kisebb vagy egyenlő	<= vagy =<
		Nem egyenlő	<> vagy ><

MEGJEGYZÉS: A számítógép tizedesvessző helyett tizedespontot használ. Ha vesszőt teszünk a számba, a számítógép két számnak tekinti (vesszővel elválasztva).

TÖRTEK ÉS TIZEDESTÖRTEK

Írhatunk egy törtet így is: .5
meg így is: 1/2
a számítógép elvégzi az osztást.
Ha osztást írunk egy PRINT utasításba,. az eredményt mindig tizedes törtként, vagy egész számként kapjuk. Például:

```
PRINT 139/493+ 5          RETURN
5.28194726
```

A π BILLENTYŰ

Álljon itt egy példa, mely a π-t használja (3.14159265...), ami a kör kerületének és az átmérőjének az arányát jelenti. Ezt az értéket a π billentyű (C= és =) lenyomásával írhatjuk be.

```
PRINT π/374          RETURN
8.39998036E- 03
```

TUDOMÁNYOS JELÖLÉS

Mit jelent a számítógép számára a fenti szám E– 03 része? A számítógép a decimális számokat a – 999 999 999 és a 999 999 999 közötti tartományban természetes számokként írja ki. Az ezen a tartományon kívül eső számokat (melyek 9 számjegynél hosszabbak) automatikusan tudományos jelöléssel jelzi ki. Magunk is beírhatunk ily módon számokat, és a számítógép gond nélkül fogja azokat értelmezni (feltehetően kevesebb gonddal, mint amennyibe nekünk került a számok konvertálása). A tudományos jelölés gyakran hasznos, mivel lehetővé teszi nagy számok kevesebb számjeggyel történő kijelzését. Lássuk, hogyan jelenik meg 198 505 478 tudományos jelöléssel: A tizedes pont bal oldalára csak egy számjegy kerül:

1.98505478E+ 8

Ez a szám a helyértékek száma, amivel a tizedespont helye megváltozott.

Egynél kisebb szám esetén, amely több tizedes jegyet is tartalmaz, a második szám negatív lesz, jelezve, hogy a tizedespont jobbra tolódott.
Például:

.0003359 = 3.359E- 4

Más példák:

20 = 2E+ 1 A tizedespont egy jeggyel balra tolódott
105000 = 1.05E+ 5 A tizedespont 5 jeggyel balra tolódott
.0666 = 6.66E– 2 A tizedespont 2 jeggyel jobbra tolódott

MŰVELETVÉGZÉS

Műveletek elvégzéséhez írjuk be a PRINT utasítást, majd a matematikai kifejezést idézőjelek nélkül, ahogy a következőkben látjuk:

```
10 PRINT 1+ 2, 2– 1
20 PRINT 2*2, 4/2
RUN
3       1
4       2
```

Az első eset, hogy a PRINT nem pontosan ugyanazt írja ki, amit az utasításba beírtunk. Ehelyett a számítógép elvégezte a számítást, és csak az eredményt írta ki. Nekünk mindössze a PRINT utasításban kellett az idézőjeleket elhagynunk. Próbáljuk ki a következőt:

```
NEW
10 PRINT "2001/2010"
20 PRINT 2001– 17
RUN
2001/2010
 1984
```

egy szóköz maradt az eredmény előjelének

Mivel a 10-es sorban idézőjel állt, a számítógép a feladatot úgy tekinti, mintha szövegről lenne szó: pontosan azt írja ki, ami az idézőjeleken belül volt.
Figyeljük meg, hogy a 10-es sorban nem maradt szóköz a szám előjelének, míg a 20-as sor eredményénél maradt.
Térjünk vissza a kurzorral a 10-es sorba, és változtassuk meg a sort a következőképpen:

```
10 PRINT "2001/2010= ";
2001/2010
RUN
```

Ne feledkezzünk meg a pontosvesszőről!

2001/2010 = .995522388 — A szóköz az eredmény előjelének helye

1984 — A 20-as sor eredménye ugyanaz maradt

Ha a feladatot is és az eredményt is ki akarjuk íratni, kétszer kell az utasításba beírnunk: először idézőjelekben, majd anélkül, ahogy az a 10-es számú sorban látható.

A MŰVELETEK SORRENDJE

Egy sorban több művelet is lehet. Próbáljuk ki a következőt:

```
PRINT 200*5+ 5
```

Az az eredmény, amire számított? Próbáljuk meg a következőt:

```
PRINT 50+ 5*200
```

A Commodore 16 a műveleteket mindig egy bizonyos sorrendben hajtja végre. A műveletek végrehajtása balról jobbra történik, de ezen belül egyes műveletek végrehajtására előbb kerül sor. Azt a sorrendet, amelyben a számítógép végrehajtja a kifejezéseket, végrehajtási vagy prioritási sorrendnek nevezzük.

ELŐSZÖR: a hatványozásra kerül sor.
MÁSODSZOR: a számítógép az előjelváltást végzi el (nem a kivonást, csak az előjelváltást).
HARMADSZOR: végrehajtja a szorzást és osztást, balról jobbra.
NEGYEDSZER: az összeadást és kivonást balról jobbra.

MEGJEGYZÉS: A Commodore 16 először mindig a zárójelen belüli kifejezést hajtja végre. Zárójeleket egymásba is skatulyázhatunk: 36*(12+(A/3)). A legbelső zárójelben lévő kifejezés végrehajtására kerül sor először. Ajánlatos a negatív számokat az érthetőség érdekében zárójelbe tenni. Például ha 45-öt mínusz 5-tel szorzunk, írjuk a következőképpen: 45*(– 5). A számítógépnek mindegy, hogy zárójellel vagy anélkül írtuk.

A KÉPERNYŐRE VALÓ NYOMTATÁSRÓL RÉSZLETESEBBEN

Valószínűleg feltűnt már, hogy egyes programokban vesszőt írtunk bizonyos helyekre, míg másokban pontosvesszőt. A számítógép a vesszőt és pontosvesszőt a nyomtatási kép elrendezésére vonatkozó utasításként értelmezi.
A PRINT utasításban lévő írásjelek hatása a szóközökre idézőjelbe tett szövegek (úgynevezett karaktersorozatok) és kiszámított eredmények esetén azonos. Próbáljuk ki a következő rövid programot:

```
NEW
10 PRINT "O", "K"
20 PRINT "O"; "K"
```

Jegyezzük meg, hogy az írásjelek az idézőjelen KÍVÜL jelennek meg!
Ha lefuttatjuk a programot, a képernyőn a következők láthatók:

```
O       K       A 10-es sor ezt nyomtatja
OK              A 20-as sor ezt nyomtatja
```

Ha a 10-es és 20-as sorok közel azonosak, miért van ilyen nagy különbség a nyomtatási képben? Az egyetlen különbség az adatok közötti írásjel.
Ha a PRINT utasításban az elemek elválasztására vesszőt használunk, az elemek néhány szóköz távolságra jelennek meg egymástól. Ha az elválasztó jel pontosvessző, az elemek egymás mellé kerülnek.

Mint nyilván emlékszünk rá, a számítógép képernyője 40 oszlop széles. A képernyő négy 10 szóköznyi széles zónára van osztva, melyeket NYOMTATÁSI ZÓNÁK-nak nevezünk. Ha vesszőt alkalmazunk a kinyomtatandó elemek elválasztására, a Commodore 16 az első tételt az első zónába, a másodikat a másodikba nyomtatja, és így tovább. A vesszők hatása az írógép tabulátorához hasonlítható.

```
1. ZÓNA                          2. ZÓNA
1 2 3 4 5 6 7 8 9 10             11 12 13 14 15 16 17 18 19 20
O                                K
3. ZÓNA                          4. ZÓNA
21 22 23 24 25 26 27 28 29 30    31 32 33 34 35 36 37 38 39 40
                                 Oszlopok
```

Ha négynél több elemet próbálunk vesszővel elválasztani, a számítógép **automatikusan a** következő sorba fog nyomtatni.

Például:

PRINT "A", "B", "C", "D", "E", "F"

a nyomtatási kép a következő lesz:

	1	11	21	31	40. Oszlop
1. sor	**A**	**B**	**C**	**D**	
2. sor	**E**	**F**			

Ha pontosvesszőt használunk a PRINT utasításban az elemek elválasztására, a Commodore 16 figyelmen kívül hagyja a nyomtatási zónákat, és az elemeket egymás mellé írja:

PRINT "A"; "B"; "C"; "D"; "E"; "F"

Ez fog a képernyőn megjelenni:

ABCDEF

Ha az első elem 12 karakter hosszú és a második vesszővel van elválasztva, az eredmény a következő:

PRINT "ABCDEFGHIJKL", "M"

1. ZÓNA 2. ZÓNA 3. ZÓNA
ABCDEFGHIJKL M

MEGJEGYZÉS: Néha különlegesen hosszú sort írunk a számítógépbe, mint például a következő:

10 PRINT "SZERETEM, HA BILLENTYŰZÖL RAJTAM, GYAKRAN CSINÁLOD?"

Ha ezt beírjuk, kifutunk a sorból. Gépeljünk azonban tovább, a számítógép automatikusan a következő sorra tér át, és folytatja az írást, amíg be nem fejezzük a sort. Egy programsorba nyolcvan karaktert írhatunk (két teljes 40 oszlopos sort). Futtassuk le ezt az egysoros programot. Az üzenet két sorban jelenik meg.
Ha a kiírandó szöveg nem fér el egy sorban, a Commodore 16 a következő sorra tér át. A számítógép a sor végét ott érzékeli, ahol a RETURN billentyűt megnyomjuk, és nem ott, ahol az általunk beírt sornak vége van. Hozzá fogunk szokni a Commodore 16 használata közben.

VÁLTOZÓK HASZNÁLATA

A következő példa a számítógép egyik leghatékonyabb tulajdonságát szemlélteti: 36*(12+ (A/3)). Ha egy matematikai kifejezésben szám helyett betű áll, azt a gép változónak tekinti. A változó egy értéket képvisel:

```
10 A = 3
20 PRINT "A SZORZAT:"; A * 4
```

Ha a programot lefuttatjuk, a képernyőn a következő jelenik meg:

```
A SZORZAT: 12
```

Háromféle változót használhatunk:

TÍPUS	JEL	LEÍRÁS	PÉLDÁK	ÉRTÉKEK
lebegőpontos		valós (decimális) vagy egész számok	X, AB, T4	23.5, 12 1.3E+ 2
egész (integer)	%	egész számok	X%, A1%	15,102,3
karakter sorozat	$	betűk, számok, és bármilyen karakter idézőjelben	X$, MS$	"TELJES", "NAP 1", "ABC"

Ha egy változót egész számként akarunk kezelni, a változó neve után % jelet kell írnunk. A szöveget tartalmazó változónak $ jelre KELL végződnie. Ha nem így végződik, a számítógép lebegőpontos számként kezeli. Ha a változó nem végződik sem %, sem $ jelre a gép lebegőpontos számként fogja értelmezni (közönséges szám). Az egész változók olyan számokat képviselnek, melyeknek nincs tört értéke. Mindig ügyeljünk a megfelelő változó típus használatára! Ha például egy szót szeretnénk megfeleltetni egész változónak, a program nem fogja ezt végrehajtani. A következő program azt szemlélteti, hogy hogyan lehet egy változót egy adott helyzetben használni, és mi történik, ha különböző típusú adatokat próbálunk beírni:

```
10 PRINT "ÍRJON BE EGY SZÁMOT"
20 INPUT X%                          a változónak egy számértéket ad
30 PRINT "OLVASOM A BEÍRT SZÁMOT";X%
40 PRINT "POMPÁSAN MEGY"
50 END
```

RUN után próbáljuk meg a következő adatokat beírni (minden beírás után újból RUN), és figyeljük meg, hogy mi történik:

EGYÖTÖD
.043
10

NUMERIKUS FÜGGVÉNYEK

A számítógép BASIC nyelve numerikus függvényeket is tartalmaz, melyek a fejlettebb tudományos célú zsebszámológépeken is megtalálhatók (ilyenek a szinusz, koszinusz, tangens, stb.). A függvény neve után a kiszámítandó értéket zárójelbe írjuk:

FÜGGVÉNY(X)

Egy változó szinuszának kiszámításához a következőket kell beírni:

PRINT SIN(X)

X egy tetszés szerinti számértéket jelöl.
A függvényeket programba is írhatjuk a következő példa alapján:

```
10 FOR X= 1 TO 5
20 PRINT"NÉGYZETGYÖK"; X; "A KÖVETKEZŐ:"; SQR(X)
30 NEXT X
```

A BASIC ismerettár tartalmazza a numerikus függvények teljes listáját.

MEGJEGYZÉS: A legtöbb parancs helyett használhatjuk annak rövidítését is. A Commodore 16 ugyanúgy kezeli a rövidítéseket, mintha a teljes utasítást írtuk volna be. A következő példa egy rövidítés használatát szemlélteti:

L majd SHIFT és O L┌ -ként jelenik meg a képernyőn.

A számítógép ezt a rövidítést LOAD-ként értelmezi. A rövidítésekkel időt takaríthatunk meg. A rövidítések teljes listája a BASIC ismerettárban található.

FELHASZNÁLÓ ÁLTAL DEFINIÁLT FÜGGVÉNYEK

A számítógép matematikai képességei hatékonyan használhatók ki a felhasználó által definiált függvények alkalmazásával. A felhasználó által definiált függvények különösen hasznosak számításokban, és könnyen alkalmazhatók a Commodore 16-on. A felhasználó által definiált függvények lehetővé teszik képletek programozását, és a kiszámítandó értékek behelyettesítését. A módszer különböző célokra használható.

A következő utasítás egy függvényt definiál, amely egy érték szekánsát adja:

```
10 DEF FNS(X)= 1/COS(X)
```

A függvény X értékéül adott bármely szám szekánsát számítja ki.
FNS az ezen utasítás által definiált függvény neve. A C függelékben találjuk meg azon trigonometrikus függvények táblázatát, amelyek nem szerepelnek a számítógép BASIC nyelvében.
A felhasználó által definiált függvények memóriát takarítanak meg, ha egy függvényt többször használunk, és a programot könnyebben olvashatóvá és érthetőbbé teszik.

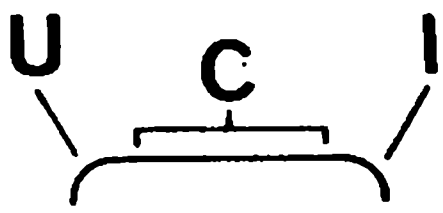
6. FEJEZET

Grafika és szín

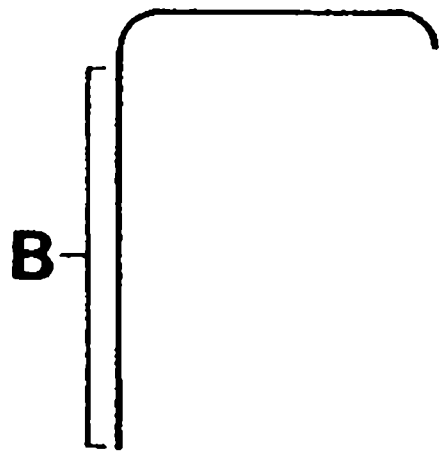
- Grafikus karakterek
- Karakter animáció
- Színek kezelése
- Nagy felbontóképességű grafika
- Pontok, vonalak és feliratok
- Négyzetek, körök, sokszögek és festés
- Többszínű grafika

GRAFIKUS KARAKTEREK

A második fejezetben említettük, hogy minden betű, valamint a @, −, +, * és £ jelek billentyűinek előlapján két különböző grafikus karakter látható a 62 különböző grafikai jel ábrázolására. A grafikus karakterek megjelenítéséhez a SHIFT vagy C= billentyűket lenyomva kell tartani miközben a kívánt grafikus jelet ábrázoló billentyűt leütjük.
Ha a Commodore 16 nem írógép üzemmódban van, a billentyűk jobb oldalán lévő grafikus karakter megjelenítéséhez nyomjuk le a SHIFT billentyűt, az adott billentyűvel együtt. A karakterek között megtalálhatók a francia kártya színei, egy kör és egy körlap, valamint vonalak és összekötő jelek, melyek segítségével különböző ábrákat rajzolhatunk a képernyőre. Ha a C= billentyűt egy betű billentyűvel együtt nyomjuk meg, a betű bal oldalán látható grafikus jelet kapjuk. A bal oldali grafikus jelek sávok, négyzetek, vonalak és blokkok választékából állnak, amelyek táblázatok és grafikonok rajzolására alkalmasak.
Vegyünk egy tetszés szerinti kártyalapot!
A következő példa alapján követhetjük, hogyan lehet a grafikus jelek segítségével ábrákat előállítani. Kövessük az utasításokat egy kártyalap, a kőr hatos megrajzolásához.
Először változtassuk a kurzor színét pirosra! Tartsuk lenyomva a CTRL billentyűt a "3"-as billentyűvel együtt a kurzor színének megváltoztatásához!

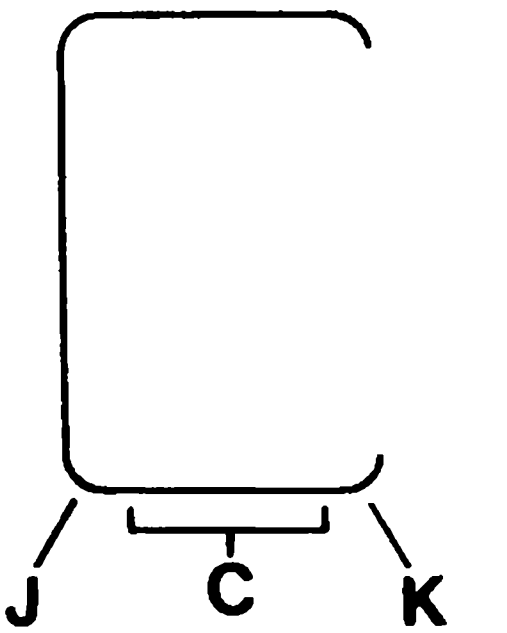


Ezután a kártya felső szélét rajzoljuk meg: nyomjuk meg a SHIFT LOCK billentyűt, hogy az lenyomva maradjon! Nyomjuk meg az "U"-t, majd ötször a "C"-t és végül az "I"-t!

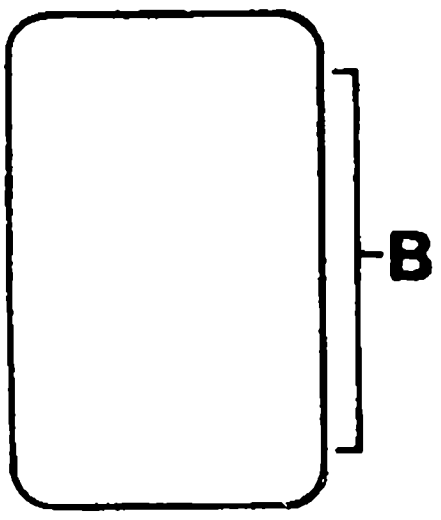


A bal oldali rajzolása: maradjon a SHIFT LOCK továbbra is lenyomva, **menjünk közvetlenül** a bal oldali ív alá (ahol az "U" billentyűt megnyomtuk **az előző lépésben**). Nyomjuk meg a "B" billentyűt, azután vigyük a kurzort **közvetlenül az éppen** berajzolt vonal alá, és nyomjuk meg újra a "B" bil-

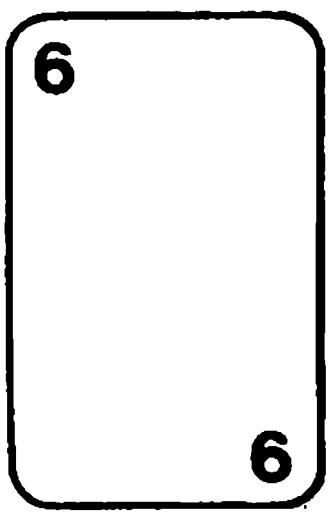
lentyűt! A "B" billetyűt így összesen 7-szer kell leütnünk. (A kurzor billentyűket ugyan most használtuk először, de gyakorlattal jobban fog menni.)



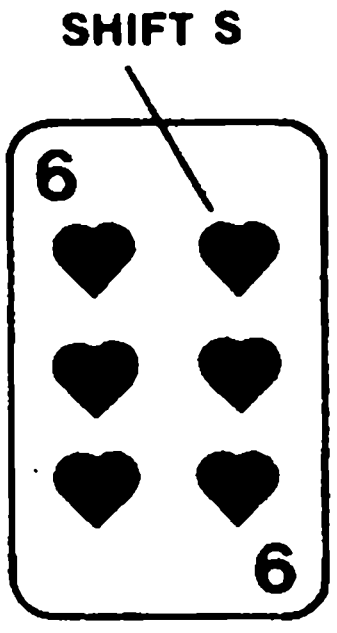
Az alsó szél rajzolása: a felső szélhez hasonlóan nyomjuk meg a "J" billentyűt egyszer, a "C" billentyűt ötször, és fejezzük be a "K" billentyűvel.



A jobb szél rajzolása: ugyanúgy végezzük, mint a bal szél rajzolását a "B" billentyű 7-szeri leütésével (miközben a kurzorral a megfelelő helyre lépünk).



Engedjük föl a SHIFT LOCK billentyűt annak újbóli megnyomásával! Írjunk egy "6"-ost a bal felső sarokba! (Egy sorral lejjebb az ívnél és egy hellyel jobbra tőle.) Lépjünk a jobb alsó sarokba, állítsuk a kurzort egy sorral feljebb és egy hellyel balra, és írjunk be egy újabb "6"-ost!



Térjünk a dolog szívéhez – 6 kell belőlük!
Nyomjuk le újból a SHIFT LOCK billentyűt! Menjünk egy sorral lejjebb, és egy hellyel jobbra attól, ahol a "6"-ost leütöttük. Nyomjuk le az "S" billentyűt, üssük le a szóközt, és nyomjuk meg újra az "S" billentyűt. Menjünk a kurzorral két sorral lejjebb, és ismételjük az "S" – szóköz – "S" sorrendet! Írjuk be az utolsó két szívet két sorral lejjebb. Ezzel elkészült szabályos kőr hatos kártyalapunk.
Huszonegyezés közben persze nem sokra mennénk vele. Gyakorlásképpen készítsünk önállóan néhány kártyalapot!

MEGJEGYZÉS: Ha a SHIFT LOCK le van nyomva, megnyomhatjuk a RETURN billentyűt, és nem kapunk SYNTAX ERROR (szintaktikai hiba) üzenetet a számítógéptől. Még akkor sem, ha a sor olyan karaktereket tartalmaz, melyeket nem lehet utasításként értelmezni. Ha a SHIFT billentyű le van nyomva, a számítógép a RETURN billentyű hatására csupán a következő sor elejére lép, a számítógép a beírt szöveget nem próbálja BASIC utasításként értelmezni.

A grafikus billentyűket a nyomtatási kép hangsúlyozására is használhatjuk. Egy szót vagy karaktert például így húzhatunk alá: vigyük a kurzort az aláhúzandó sor alá! Tartsuk lenyomva a C= billentyűt és a "T"-t mindaddig, míg a teljes szövegrészt alá nem húzzuk!
Ennek a résznek az a célja, hogy bemutassa, hogyan lehet a Commodore 16 grafikus jeleit különböző formák és ábrák létrehozására célszerűen alkalmazni. A rendelkezésünkre álló 62 grafikus karakteren kívül ezek inverz változatát is használhatjuk. (Emlékezzünk a versenypályára!)
Miután megismertük a különböző ábrák előállításához szükséges grafikus jelek alkalmazásának módszerét, már csak a gyakorlat hiányzik az alkalmazásukhoz.

KARAKTER ANIMÁCIÓ

A mozi valójában állóképek sorozata. Mindegyik kép kissé különbözik az előtte lévőtől. A vetítő minden képet csak nagyon rövid ideig mutat, majd a következőre ugrik. A jelenet így életre kel.
A számítógépes animáció hasonlóan működik. A számítógép először rajzol egy képet, majd kissé megváltoztatja. A Commodore 16 elég gyors ahhoz, hogy a tárgyakat folyamatosan mozgassa az ernyőn a játékokban, vagy a gyakorlati alkalmazás során. A moziban másodpercenként 24 képet vetítenek. Ez a változás elég gyors ahhoz, hogy megtévessze a szemet, és mozgás illúzióját keltse.
Számítógépen ezt programmal érhetjük el, mely megjelenít egy képet, várakozik a másodperc tört részéig, majd új képre vált. A képet létrehozó programban a PRINT utasítást használjuk a grafikus karakterekkel.
Az animáció legegyszerűbb módja két karakter váltogatása a mozgás hatásának érzékeltetésére.
Ez a program az animációt a kör (SHIFT és Q) és a szív (SHIFT és S) jelek váltogatásával valósítja meg. Kis fantáziával ezt valamiféle szívverésnek tekinthetjük.

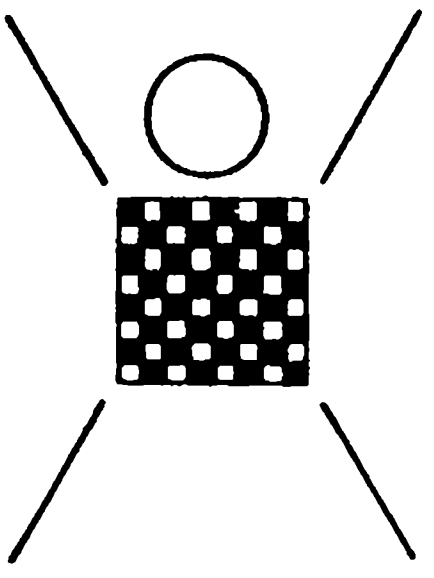
FONTOS MEGJEGYZÉS: A SHIFT vagy C= billentyűket minden esetben az utánuk következő billentyűvel EGYÜTT kell lenyomni a programírás közben mivel önmagukban leütve hatástalanok.
Ne felejtsük el beírni a NEW parancsot, és megnyomni a RETURN billentyűt, mielőtt új programot írunk be! A progamsorok beírását is RETURN-nel fejezzük be!

```
10 PRINT "HOME SHIFT Q"
20 FOR L= 1 TO 100
30 NEXT L
40 PRINT "HOME SHIFT S"
50 FOR M= 1 TO 200
60 NEXT M
70 GOTO 10
```

Ezeket egyszerre nyomjuk le

Két karakter váltogatása még nyilvánvalóan korlátozott animációt eredményez.
A program leállításához nyomjuk meg a RUN/STOP billentyűt. Érdekesebb hatást érhetünk el, ha különböző grafikus karakterekből ábrát alkotunk, és egyes jeleket megváltoztatunk, míg másokat változatlanul hagyunk. Ez olyan hatást kelt, mintha egy nagyobb tárgy egy része mozogna.
Ezt szemlélteti a következő program:

```
10 PRINT"HOME SHIFT M SHIFT W SHIFT N"
20 PRINT"C= + "
30 PRINT" SHIFT N SHIFT M"
40 FOR L= 1 TO 300:NEXT L
50 PRINT "HOME SHIFT W"
60 PRINT "C= T C= + C= T"
70 PRINT "C= G C= G"
80 FOR L= 1 TO 300: NEXT L
90 GOTO 10
```



Írjuk be a RUN parancsot, és nyomjuk meg a RETURN billentyűt! Mindkét animációs példában eddig a képek a képernyő ugyanazon helyén álltak. A következő lépés az animációs képek mozgatása lesz. A TAB függvény alkalmas a tárgyak balról jobbra való mozgatásához. (A TAB függvény részletes leírása a BASIC ismerettárban található.)
A következő program kígyózó mozgást ábrázol a képernyőn. Ne feledjük, hogy a SHIFT-et és az azt követő billentyűt egyszerre kell lenyomni!

```
5 FOR A= 0 TO 30
10 PRINT "SHIFT CLR"
20 PRINT TAB (A) "SHIFT U SHIFT I SHIFT U SHIFT I"
30 PRINT TAB (A) "SHIFT K SHIFT J SHIFT K SHIFT J SHIFT Q"
40 FOR L= 1 TO 100: NEXT L
50 PRINT "SHIFT CLR"
60 PRINT TAB(A+ 1)"SHIFT I SHIFT U SHIFT I SHIFT U SHIFT W"
70 PRINT TAB(A+ 1)"SHIFT J SHIFT K SHIFT J SHIFT K"
80 FOR L= 1 TO 100: NEXT L
90 NEXT A
```

A 20-as és 30-as sorok ilyen képet adnak:

A 60-as és 70-es sorok ilyet:

Labdához hasonló karakterek használatával (SHIFT Q) videojátékokat játszhatunk a képernyőn. A labda mozgatásához először töröljük a labdát, majd új helyen jelenítsük meg, mint az a következő programban látható:

```
10 PRINT "SHIFT CLR"
20 PRINT "SHIFT Q ←"
30 FOR L= 1 TO 50: NEXT L
40 GOTO 20
```

kurzor balra nyíl

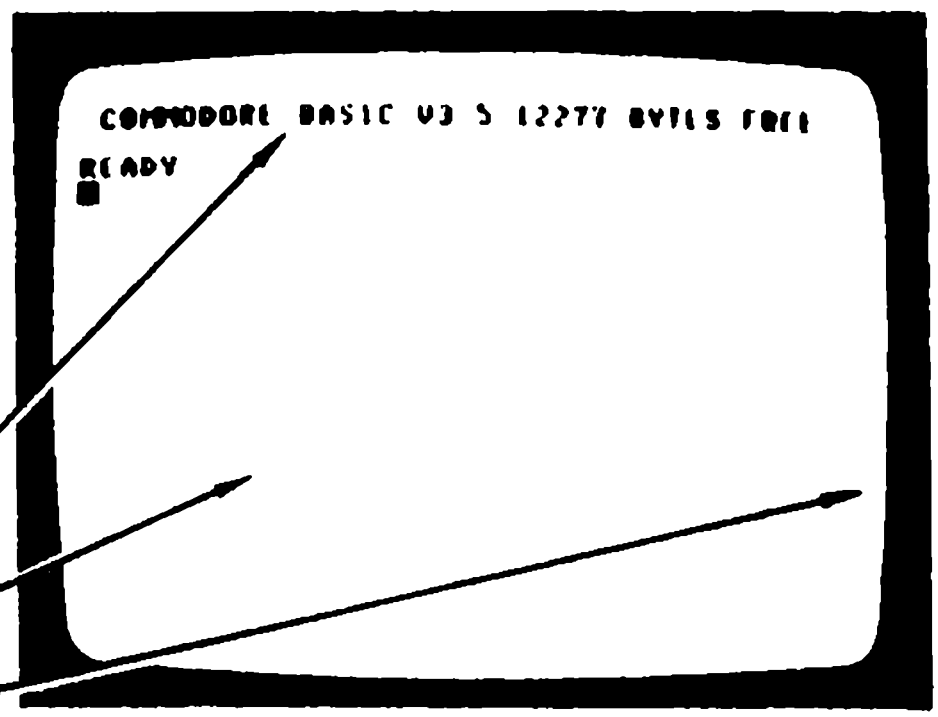
A futó programot a RUN/STOP billentyűvel állíthatjuk meg.

SZÍNEK KEZELÉSE

Különböző színeket állíthatunk be a képernyő egyes részein. Lehet a keret az egyik szín, a háttér egy másik, és minden karakternek saját színe lehet. Már tudjuk, hogyan lehet beállítani a karakterek színét a billentyűzet használatával. A képernyő egyéb részeinek színét a COLOR BASIC paranccsal változtathatjuk meg.
A képernyő keretét például a COLOR 4,3 parancs végrehajtásával és a RETURN billentyű megnyomásával változtathatjuk pirosra. A parancsban lévő 4-es a keretet jelenti, a 3-as pedig a piros színt. (A RED (piros) feliratú billentyűn lévő szám.)
Írjuk be a COLOR 0, 7 parancsot és üssük le a RETURN billentyűt! A háttér színe kékre változik. A 0 a hátteret jelenti, míg a 7-es a kék színt. (Szintén ugyanaz, mint a billentyűzeten.)
A COLOR parancs után álló első szám a változtatandó képernyőrészt jelöli ki. A következő táblázat a képernyő egyes részeinek azonosító számait tartalmazza: a 2-es és 3-as kódok használatáról többet is fogunk hallani ebben a fejezetben a többszínű grafika tárgyalásakor.

A KÉPERNYŐTERÜLETEK SZÁMAI

TERÜLET SZÁMA	TERÜLET NEVE
0	Háttér
1	Karakter
2	Többszínű 1
3	Többszínű 2
4	Keret



A COLOR után következő második szám a kijelölt képernyőterület színét határozza meg. A színt azonosító számok a billentyűzeten lévő szín billentyűknek felelnek meg 1 – 8-ig, illetve a következő oldalon levő táblázat szerintiek.

SZÍN AZONOSÍTÓ SZÁMOK

SZÍN SZÁMA	SZÍN	SZÍN SZÁMA	SZÍN
1	Fekete	9	Narancs
2	Fehér	10	Barna
3	Piros	11	Sárgászöld
4	Türkiz	12	Rózsaszín
5	Bíbor	13	Kékeszöld
6	Zöld	14	Világoskék
7	Kék	15	Sötétkék
8	Sárga	16	Világoszöld

A színeknek állítható a fényereje is. A parancsba írt harmadik paraméter, mely 0-tól (legsötétebb) 7-ig (legfényesebb) terjed, a fényerő beállítására szolgál. Írjuk be a COLOR 4,3,0 parancsot és nyomjuk meg a RETURN billentyűt! A keret sötétpiros lesz. A COLOR 4,3,7 parancs hatására a keret élénk pirosra változik. A COLOR parancs általános formája a következő:

COLOR terület, szín, fényerő

Az itt következő rövid program a Commodore 16 színeit mutatja be.
Először a NEW parancsot írjuk be, majd a RETURN billentyűt nyomjuk meg.

```
10 COLOR 0, 7, 7
20 FOR M= 0 TO 7
30 FOR N= 1 TO 2
40 FOR L= 1 TO 16
50 PRINT "CTRL RVS ON";
60 READ A
70 COLOR 1, A, M
80 PRINT "Szóköz Szóköz";
90 NEXT L
100 PRINT
110 RESTORE
120 NEXT N,M
130 COLOR 1, 2, 4
200 DATA  7,14,4,13,6,16,11,8,10,9,3,12,5,15, 2,1
```

Ezeket a billentyűket egyszerre nyomjuk meg!

Ha ezt a programot futtatjuk, a háttér világoskékre változik, és a Commodore 16 színeinek spektrumát látjuk minden fényerő fokozaton. Tapasztalni fogjuk, hogy a fekete fényereje minden fényességi szinten azonos.

MEGJEGYZÉS: A legtöbb BASIC grafikus kifejezéshez hasonlóan, amelyeket ebben a fejezetben megemlítünk, a COLOR utasításként és parancsként egyaránt használható.

A GRAPHIC PARANCS

Az eddig látott grafikák csak a billentyűzetet használták a számítógép képességeinek tényleges kihasználása nélkül. A Commodore 16 BASIC nyelve programból történő alak és forma rajzolására is tartalmaz utasításokat. A grafikus parancsok használatához új üzemmódba, a GRAFIKUS üzemmódba kell áttérni. A grafikus üzemmódot rajzoló üzemmódnak tekinthetjük, mivel az összes rajzoló parancs alkalmazása lehetővé válik. Ezek a parancsok addig nem használhatók, míg pontosan ki nem jelöljük, hogy milyen grafikus üzemmódot kívánunk használni. A grafikus üzemmód kijelölésére a GRAPHIC parancs szolgál. Három különböző üzemmód létezik: szöveg, nagy felbontóképességű grafika és többszínű grafika. A GRAPHIC paranccsal részben szöveges, részben grafikus képernyőt is definiálhatunk, mely lehetővé teszi, hogy a képernyő egyik részére írjunk, a másikra pedig rajzoljunk. Ezen új üzemmódra való áttérés a GRAPHIC paranccsal lehetséges.
A GRAPHIC parancs általános alakja a következő:
GRAPHIC üzemmód, törlés ← ez a rész opcionális

Üzemmód száma	Hatás
0	Szöveg
1	Nagy felbontóképességű grafika
2	Nagy felbontóképességű grafika + szöveg
3	Többszínű grafika
4	Többszínű grafika + szöveg

Törlés száma	Hatás
0	Nem törli a képernyőt
1	Törli a képernyőt

A szokásos grafikus (szöveg) üzemmódból a nagy felbontóképességűbe való áttéréshez írjuk be a GRAPHIC 2,1 parancsot és nyomjuk meg a RETURN billentyűt! A képernyő törlődik, és a kurzor a képernyő aljához közel jelenik meg. A Commodore 16 képernyője két különálló részre oszlik: a fölső grafikus részre, és az alsó öt sorra, mely szöveg megjelenítésére szolgál. Ha nem akarjuk az alsó öt sort szöveg megjelenítésére igénybe venni, használhatjuk a GRAPHIC 1,1, parancsot, de ebben az esetben nem látjuk amit beírunk.

A GRAPHIC parancs használatával tetszés szerint térhetünk át szöveg módból grafikusba és viszont. A GRAPHIC 0 parancs visszaállítja az ernyőt szöveg módba, míg a GRAPHIC 2 a nagy felbontóképességű üzemmódba kapcsol anélkül, hogy törölné a képernyőt. A parancs után ,1-et írva a képernyő törlődik. A nagy felbontóképességű képernyőt más módon is törölhetjük. Az SCNCLR parancs anélkül törli a képernyőt, hogy a grafikus üzemmódon változtatna.
A nagy felbontóképességű grafika használatakor a számítógép 10 K memóriát fönntart a nagy felbontóképességű képernyő számára. Ezt a memóriát a BASIC programterületből veszi el. Ha már nem használjuk a grafikát, a memóriát a GRAPHIC CLR parancs segítségével szerezhetjük vissza.

NAGY FELBONTÓKÉPESSÉGŰ GRAFIKA

A képernyő huszonöt 40 karakteres sort, azaz 1000 karaktert tartalmaz. Minden karakter egyedi pontokból épül fel, melyek 8 sorban, soronként 8 pontban helyezkednek el. Így a képernyő soronként 320 pontot tartalmaz, ami 200 sorban 64000 pontot eredményez. A nagy felbontóképességű grafika lehetővé teszi, hogy minden egyes pontot külön vezéreljünk.
Normál grafika használatakor nincs módunk az egyedi pontokat vezérelni. Versenypálya vagy kártya rajzolása jó gyakorlat, de csak a billentyűzeten lévő karakterek (betűk, grafikai jelek, stb.) használatára korlátozódik. Alkothatunk ugyan többféle alakot és ábrát, de csak a töredékét annak, amit minden egyes pont vezérlésével tehetnénk. A Commodore 16 nagy felbontóképességű grafikája éppen ezt teszi lehetővé. A felbontóképesség a rajzolás közbeni pontosságra és a pontok vezérlésére vonatkozik. Nagy felbontóképességű grafikus módban olyan parancsokat adhatunk ki, melyek lehetővé teszik pontok, vonalak, körök és egyéb formák rajzolását és törlését.
A nagy felbontóképességű grafikának van azonban egy korlátja is. A számítógép csak két szín használatát teszi lehetővé minden 8 x 8 pontból álló karakterpozícióban. Azaz minden karakterhely két színre korlátozódik (előtér és háttér színre). Minden karakterpozícióra különböző színt használhatunk, de egy pozíción belül csak két színt. A másik grafikus üzemmód, a többszínű üzemmód, melyet később tárgyalunk, lehetővé teszi 4 különböző szín használatát karakterpozíciónként a nagy felbontóképességű üzemmód rovására.

A következő program a számítógép nagy felbontóképességű grafikai képességeit és különösen a GRAPHIC parancs használatát szemlélteti. Töröljük a számítógép memóriáját a NEW paranccsal, hogy biztosan ne maradjanak programsorok a gépben, és írjuk be:

```
10 COLOR 0,1
20 GRAPHIC 1,1
30 FOR L= 2 TO 16
40 COLOR 1,L,2
50 DRAW 1,0,L*12 TO 319,L*12
60 DRAW 1,L*18,0 TO L*18,199
70 NEXT L
80 FOR L= 1 TO 5000:NEXT L
90 COLOR 1,2,3
100 GRAPHIC 0
```

Tapasztaljuk, hogy a színek megváltoznak a csomópontok közelében. Ez a nagy felbontóképességű grafika korlátai miatt van, mivel túl sok szín van egymás közelében.

PONTOK, VONALAK ÉS FELIRATOK

Írjuk be a GRAPHIC 2,1: DRAW 1,0,0 parancsokat, és nyomjuk meg a RETURN billentyűt! Nézzük meg közelről a képernyő bal felső sarkát! a Commodore 16 egy fehér pontot rajzolt ide. A DRAW parancs használható egyetlen pont vagy vonal, illetve forma bárhová történő rajzolása.

Néhány példa a DRAW parancsra:

PARANCS	EREDMÉNY
DRAW szín, oszlop, sor	Pont
DRAW szín, oszlop, sor TO oszlop, sor	Vonal
DRAW szín TO oszlop, sor	Vonal rajzolása az utolsó ponttól

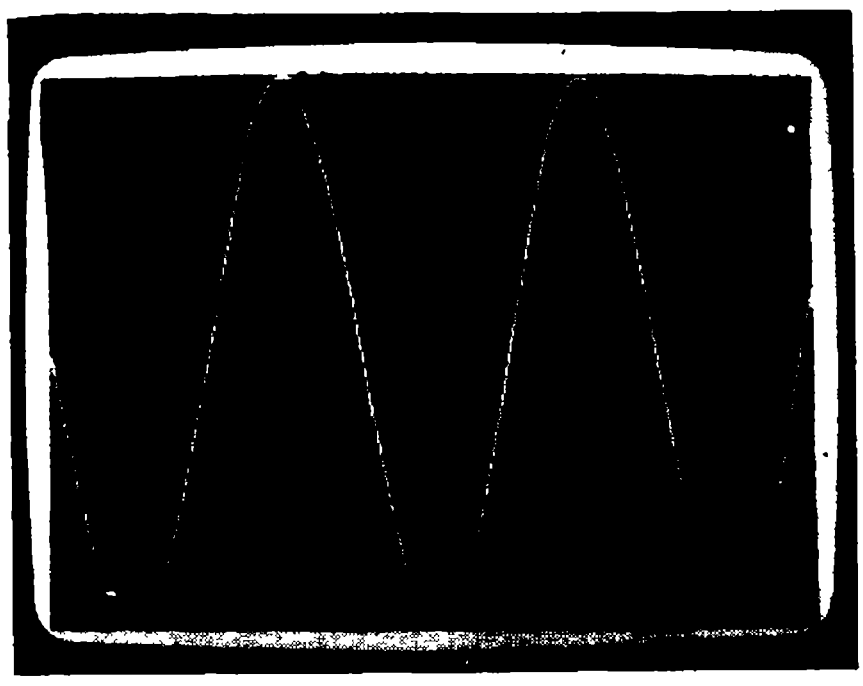
A szín 0 a háttér, és 1 az előtér esetén. A háttér színével (0) való rajzolás ugyanazt eredményezi, mint az előtér színének ugyanazon helyen való törlése.
A DRAW parancsban az első szám vagy 1 (pont rajzolása) vagy 0 (pont törlése). A következő két szám a pont sor- és oszlopszámára vonatkozik. Így ha egy pontot akarunk rajzolni a 20. sor 17. oszlopára, a következő parancsot kell beírnunk:

DRAW 1,20,17	Ugyanezen pont törlése a
DRAW 0,20,17	paranccsal lehetséges

A DRAW paranccsal vonalat is rajzolhatunk két pont közé. Csak adjuk meg a TO szócskát, és a másik vég koordinátáit a következő módon: DRAW

1,1,1, TO 100,100. Ez a parancs vonalat rajzol az 1,1 ponttól a 100,100 pontig. A vonalat ugyanezzel a paranccsal törölhetjük, csak az első 1-es helyere kell 0-át írnunk.
Ha gyakorlatunk van grafikonok rajzolásában, először megtévesztő lehet a számítógép használata. A Commodore 16 koordináta-rendszere különbözik az eddig megszokottól. A matematikában a 0,0 pont vagy a képernyő közepén, vagy a bal alsó sarkában volt, a számítógépen viszont ez a bal felső sarok. A gyakorlat során hozzá fogunk szokni ehhez a rendszerhez.
Ha már rajzoltunk egy pontot a képernyőre, vonalat húzhatunk ettől egy másik pontig a következő módon: DRAW 1 TO 150,50. A parancs vonalat rajzol az utolsó ponttól az 50. sor 150. oszlopáig. Ha programunk sok DRAW TO parancsot használ, az első pontot a LOCATE paranccsal helyezhetjük el a képernyőn, mint az alábbi példában a LOCATE 0,100. A következő program egy szinusz függvény görbéjét rajzolja a képernyőre:

```
NEW
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 1,1
40 LOCATE 0,100
50 FOR X= 1 TO 319
60 Y = INT (100+ 99 * SIN(X/20))
70 DRAW 1 TO X,Y
80 NEXT X
90 FOR L= 1 TO 5000
100 NEXT L
110 GRAPHIC 0
```



Ne írjuk be a NEW parancsot a program elindítása után. Változtassuk meg a program 70-es sorát a következő módon:

```
70 DRAW 1, X, Y
```

A program ugyanazt a görbét rajzolja, de vonal helyett pontokból.

A CHAR PARANCS

A görbék sokkal könnyebben azonosíthatók és használhatók, ha feliratozzuk őket. A CHAR parancs használatával szöveget keverhetünk a nagy felbontású rajzok közé. A CHAR 1,0,5,"SZIA" utasítás például a SZIA szót a képernyő bal szélének hatodik sorába helyezi (az első sor a 0-ás). A CHAR szó után álló első szám vagy 1-es (rajzolás) vagy 0 törlés. A következő két

szám az oszlop és sor, ahol a szöveg megjelenik. Hagyjuk az utolsó két programot a számítógépben, ne írjuk be a NEW parancsot!
Írjuk be a következő sorokat:

```
81 CHAR 1,0,0,"Y= SIN(X)":CHAR 1,0,1,"KÉPLET"
82 CHAR 1,0,2,"GRAFIKONJA"
83 DRAW 1,0, 100 TO 319,100,189,0 TO 189,199
84 CHAR 1,0,12,"X TENGELY": CHAR 1,22,0,"Y"
85 CHAR 1,22,2,"T": CHAR 1,22,3,"E"
86 CHAR 1,22,4,"N": CHAR 1,22,5,"G"
87 CHAR 1,22,6,"E": CHAR 1,22,7,"L"
88 CHAR 1,22,8,"Y"
```

NÉGYZETEK, KÖRÖK, SOKSZÖGEK ÉS FESTÉS

A DRAW parancs használatával képeket is rajzolhatunk sok pont és vonal elhelyezésével. Négyzet rajzolásához a következő parancsot használhatjuk: DRAW 1,0,0 TO 100,0 TO 100,100 TO 0,100 TO 0,0. Négy pontot négy vonallal csatlakoztatunk (a négyzet minden oldalát kirajzolva), de használhatjuk közvetlenül a BOX parancsot is.

DERÉKSZÖGŰ NÉGYSZÖGEK RAJZOLÁSA

A Commodore 16 egyik grafikai parancsa megkönnyíti a négyzetek és egyéb derékszögű idomok rajzolását. A BOX parancs lehetővé teszi, hogy egy négyzetnek csak két szemben lévő csúcsát adjuk meg. Az előbbi példa megrajzolásához elegendő a BOX 1,0,0,100,100 parancsot megadni. Az 1-es szám ismét azt jelenti, hogy rajzolni akarunk és nem törölni. A következő négy szám az alakzat szemben lévő csúcsainak koordinátái, (0,0) a bal felső csúcs és (100,100) a képernyő középpontjához közel a jobb alsó.
A BOX paranccsal tetszés szerinti derékszögű négyszöget rajzolhatunk a csúcsok koordinátáinak megváltoztatásával. Az alakzatot el is forgathatjuk egy az utolsó koordináta után megadott szöggel (fokokban megadva) a következőképpen: BOX 1,50,50,100,100,45.
Az alakzat az óramutató járásának irányában 45 fokkal elfordul. Ha a kontúr helyett tömör alakzatot akarunk rajzolni, a szög után írjunk vesszőt és egy 1-est. A képernyő közepén lévő tömör alakzat a következőképpen rajzolható: BOX 1,100,50,220,150,,1. Jegyezzük meg, hogy a szög helyére akkor is vesszőt kell tenni, ha a téglalapot nem akarjuk elforgatni. Erre azért van szükség, mert így a számítógép utasítás hiányában az ALAPÉRTELMEZÉS szerinti utasítást hajtja végre. Ha a vesszőt kihagynánk, a gép a sor végén álló 1-est forgatási szögként értelmezné.

A BOX parancs szokásos formái a következők:

PARANCS	HATÁS
BOX be,oszlop1, sor1, oszlop2, sor2	Kontúr
BOX be,oszlop1, sor1, oszlop2, sor2, szög	Elforgatás
BOX be,oszlop1, sor1, oszlop2, sor2,, kitöltés	Kitöltött alakzat
BOX ki,oszlop1, sor1, oszlop2, sor2,,szög,kitöltés	Törli a képernyőt

Az oszlop1, sor1, stb. a végpontok képernyőn elfoglalt helyét jelölik. Az oszlop1, sor1 az alakzat bal felső csúcsa, míg az oszlop2, sor2, a jobb alsó csúcs.
A következő program a BOX parancsot szemlélteti (60-as sor):

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 2,1
40 A = RND(1)* 20+ 10
50 FOR L= 0 TO 359 STEP A
60 BOX 1,100,30,220,130,L
70 NEXT L
80 FOR L= 1 TO 2000: NEXT L
90 GRAPHIC 0,1
```

KÖRÖK RAJZOLÁSA

A Commodore 16 rendelkezik kör-rajzoló parancsokkal is. A BOX parancshoz hasonlóan változtathatjuk a kör alakját is (ellipszist formálva), és el is forgathatjuk az ellipszist.
Megrajzolhatjuk az alakzat egy ívét is.
A CIRCLE parancs szokásos alakjai a következők:

PARANCS	HATÁS
CIRCLE be, középpont oszlop, középpont sor, sugár	Kör
CIRCLE be, kp oszlop, kp sor, szélesség, magasság	Ellipszis
CIRCLE be, kp oszlop, kp sor, szélesség, magasság, kezdet, vég	Ív
CIRCLE be, kp oszlop, kp sor, szélesség, magasság,,, szög	Elforgatott ellipszis
CIRCLE be, kp oszlop, kp sor, szélesség, magasság,,,, 360/szögek száma	Sokszög

A következő parancs kört rajzol a képernyő közepére: CIRCLE 1,160,100,50. Arra utasítja a számítógépet, hogy rajzoljon egy kört, amelynek középpontja a 160-as oszlop 100-as sora, és sugara 50. Lehet, hogy az eredmény ellipszis lesz, mivel néhány TV monitor (például az amerikaiak) képpontjainak magassága nagyobb a szélességénél.

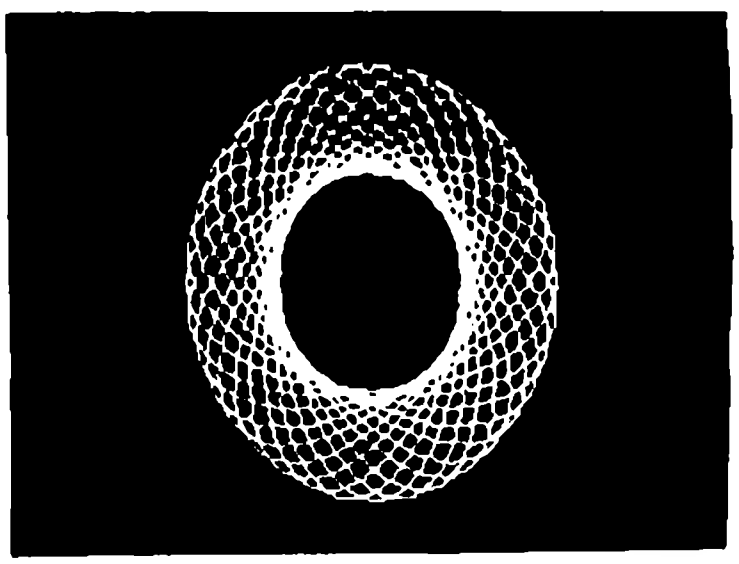
SOKSZÖGEK RAJZOLÁSA

A Commodore 16 négyzetet, háromszöget vagy egyéb sokszöget is tud rajzolni a CIRCLE paranccsal. Csak azt kell közölnünk, hogy az idom csúcspontjai hány fokra vannak egymástól: CIRCLE 1,160,100,50,42,,,,120. Ez a parancs háromszöget rajzol, mivel csúcsai 120 fokra vannak egymástól. Emlékezzünk, ha egyes paramétereket vesszővel helyettesítünk, helyettük a számítógép az alapértelmezés szerinti értéket alkalmazza. (N oldalú sokszög szögeit az egyszerű 360/N képlettel számíthatjuk.)
Példa sokszögek rajzolására:

```
10 GRAPHIC 2,1
20 INPUT"HÁNY OLDAL";A
30 IF A<2 OR A>100 THEN PRINT"NE LÉGY NEVETSÉGES": GOTO 20
40 CIRCLE 1,160,80,40,3,,,,360/A
50 GOTO 20
```

Teljes kör helyett rajzolhatunk körívet is. A CIRCLE parancsba az ív kezdetét és végét fokokban kell beírni közvetlenül a magasságot megadó paraméter után.
A CIRCLE 1,160,100,50,42,90,180 parancs a körnek csak a jobb alsó ívét jeleníti meg.
Ellipszis elforgatásához az elforgás szögét az óramutató járásának megfelelően a következő példa szerint kell megadni:
CIRCLE 1,160,100,100,20,,,30
A következő program a CIRCLE parancs segítségével érdekes hatást nyújt. Ne felejtsük el a memóriát törölni a NEW parancs beírásával!

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 1,1
40 A = RND(1)* 20+ 10
50 FOR L= 0 TO 359 STEP A
60 CIRCLE 1, 160, 100, 80, 40,,,L
70 NEXT L
80 FOR L= 1 TO 2000:NEXT L
90 GRAPHIC 0,1
```

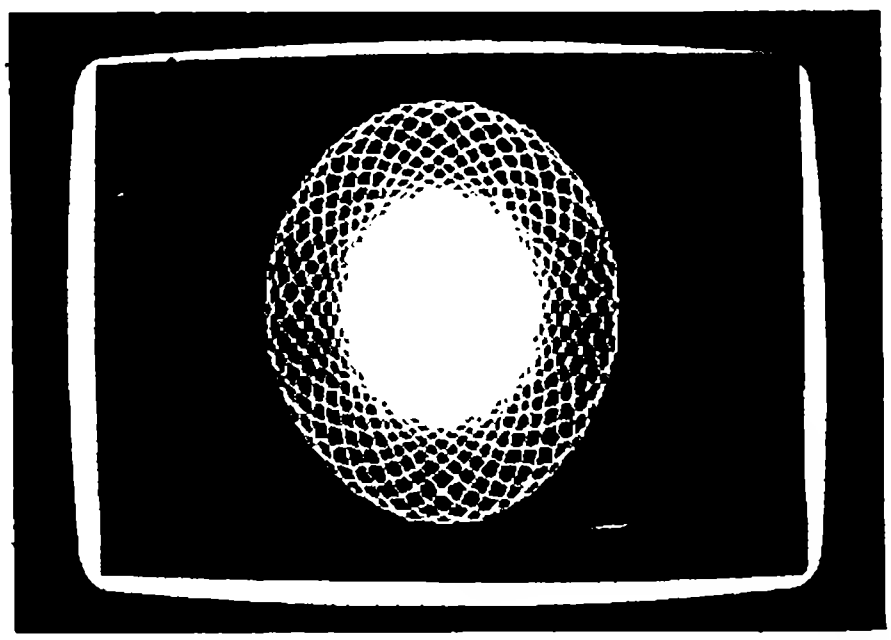


A program minden futása az előzőtől kis mértékben eltérő képet eredményez.

A PAINT PARANCS (FESTÉS)

Ha egy körnek vagy egyéb alakzatnak nemcsak a kontúrját akarjuk megrajzolni, a PAINT parancsot használhatjuk. A PAINT parancs bármely zárt területet kitölt a vonalak által formált határokig. Ha a terület nem zárt, a kitöltés a képernyő széléig terjed. A BOX parancsban is ki lehet tölteni a derékszögű négyszögeket. A PAINT paranccsal szabálytalan formákat és egyéb nem szabályos területeket is kiszínezhetünk, melyek más paranccsal nem tölthetők ki.
A PAINT parancs működésének szemléltetésére adjuk a következő sort az utolsó programhoz:

75 PAINT 1, 160, 100



TÖBBSZÍNŰ GRAFIKA

A Commodore 16 nagy felbontóképességű grafikája lehetővé teszi a képernyő minden egyes pontjának vezérlését, de tapasztalhattuk, hogy a színek használata korlátozott. A legtöbb nagy felbontóképességű program csak egy vagy két színt használhat. Több különböző szín alkalmazásához a számítógép egy "közbülső" grafikai üzemmódba helyezhető, amit többszínű grafikus módnak nevezünk. Többszínű grafika esetén soronként fele annyi pontot kezelhetünk, mint nagy felbontóképességű grafika esetén, mivel minden pont kétszer olyan széles. 200 sor, és soronként 160 pont áll rendelkezésünkre. Ezáltal lehetővé válik több szín használata, míg a felbontás kissé rosszabb. A többszínű grafika alkalmazásához tekintsük át a GRAPHIC parancs használatát! A többszínű képernyő szöveg nélkül a GRAPHIC 3, míg 5 sor szöveggel a GRAPHIC 4 paranccsal állítható be.
Tekintsük át a COLOR parancs paramétertáblázatát! Két paramétert nem használtunk eddig, a 2-est és a 3-ast. Ezekkel két további szint alkalmazha-

tunk. Ily módon három szín áll rendelkezésünkre, (1 a szövegszín; 2 és 3 két további szín). Ezek a színek nem hatnak egymásra, mint azt a nagy felbontóképességű grafika alkalmazásakor néhány előző programban tapasztaltuk.
A következő program többszínű grafikát alkalmazva "neonreklám" hatást szemléltet:

```
NEW
10 COLOR 0,1
20 GRAPHIC 3,1
30 COLOR 3,1
40 TRAP 200
50 DRAW 3,10,10 TO 10,100: DRAW 3,10,55 TO 30,55
60 DRAW 3,30,10 TO 30,100: DRAW 3,50,10 TO 80,10
70 DRAW 3,65,10 TO 65,100: DRAW 3,50,100 TO 80,100
80 FOR L= 0 TO 7
90 COLOR 3,2,L
100 FOR M= 1 TO 100: NEXT M
110 NEXT L
120 COLOR 3,1
130 FOR M= 1 TO 100: NEXT M
140 GOTO 80
200 GRAPHIC 0: COLOR 1,2,7
```

A 3-as színterület, a többszínű területek közül a második, különleges képességgel bír, mellyel a többiek nem rendelkeznek. Ha a 3-as terület felhasználásával rajzolunk a képernyőre, a képernyőn lévő színt bárhol megváltoztathatjuk a COLOR paranccsal. Ha a színt a COLOR 3,5 paranccsal állítjuk be, és ezzel rajzolunk, az ábra bíbor színben jelenik meg. Ha ezután megváltoztatjuk a színt a COLOR 3,6 paranccsal, minden bíbor terület zöldre változik. Ez nem következik be egyetlen más terület esetén sem.

7. FEJEZET

Hang és zene

- Bevezetés
- A Volume parancs (hangerő)
- A Sound parancs (hang)
- Hanghatások létrehozása
- Zene megszólaltatása
- A zenegép

BEVEZETÉS

A következő rövid program zene megszólaltatását szemlélteti a Commodore 16-on. Begépelés és RUN után egy kérdőjel jelenik meg a képernyőn. Írjunk be egy tetszés szerinti számot 0-tól 1015-ig, és nyomjuk meg a RETURN-t! A program megállításához írjunk be 0-át.

```
10 VOL 8
20 DO
30 INPUT X
35 IF X>1015 OR X< 0 THEN PRINT "0 ÉS 1015 KÖZÖTT KÉREM A SZÁMOT": GOTO 30
40 SOUND 1, X, 10
50 LOOP UNTIL X= 0
```

A RUN/STOP billentyű megnyomása szintén megállítja a programot.
A Commodore 16-on egy hangjegyet a következőképpen lehet megszólaltatni:
Először: írjuk be a VOL 8 parancsot és nyomjuk meg a RETURN billentyűt!
Másodszor: írjuk be a SOUND 1,266,60 parancsot és nyomjuk meg RETURN billentyűt!
Egy hangot kellene hallanunk, ami körülbelül egy másodperc múlva abbamarad. Tekinthetjük ezt egy befejezetlen szimfóniának a szó legszélsőségesebb értelmében (egy hang már kész, már csak 3500 hiányzik). Ha nem hall semmit, erősítse fel a televízió vagy monitor hangerejét, és próbálkozzon újra.
Csupán ezt a két parancsot kell ismerni, hogy zenét játszhassunk a Commodore 16-on. Mindkét parancs könnyen érthető, és még könnyebben alkalmazható.

A VOLUME PARANCS (HANGERŐ)

A VOL parancs a Commodore 16-on lejátszandó hang hangerejét szabályozza. A VOL parancs után álló szám a hangerő beállítása. Ez a parancs úgy működik, mint a televízió hangerő gombja. Ha nullára állítjuk (VOL 0), a hangot kikapcsoljuk és semmit sem hallunk. Ha nyolcra állítjuk (VOL 8), a hangerő maximális, és a számítógép olyan hangosan játszik, ahogy tud.
Próbáljuk ki újra az első példát különböző számokkal a VOL parancs után! Minél nagyobb a szám, annál hangosabban hallatszik a hang.

A SOUND PARANCS (HANG)

A SOUND parancs közli mindazt a számítógéppel, amit a hang lejátszásáról tudni kell. A SOUND parancsot három szám követi, melyek a hangot leírják:

SOUND szólam, hangmagasság, időtartam

A SOUND parancsban álló első szám, mely 1, 2 vagy 3 lehet, a szólamokra vonatkozik. A Commodore 16 két különböző szólamot képes megszólaltatni. Egyest használunk az első szólam, kettest a második megszólaltatásához. Hármassal a második hangforrást vezéreljük hanghatás vagy zaj létrehozására.

Első szólam – Csak zenei hangokat állít elő. Kiválasztásához a SOUND parancs után egyest írunk.

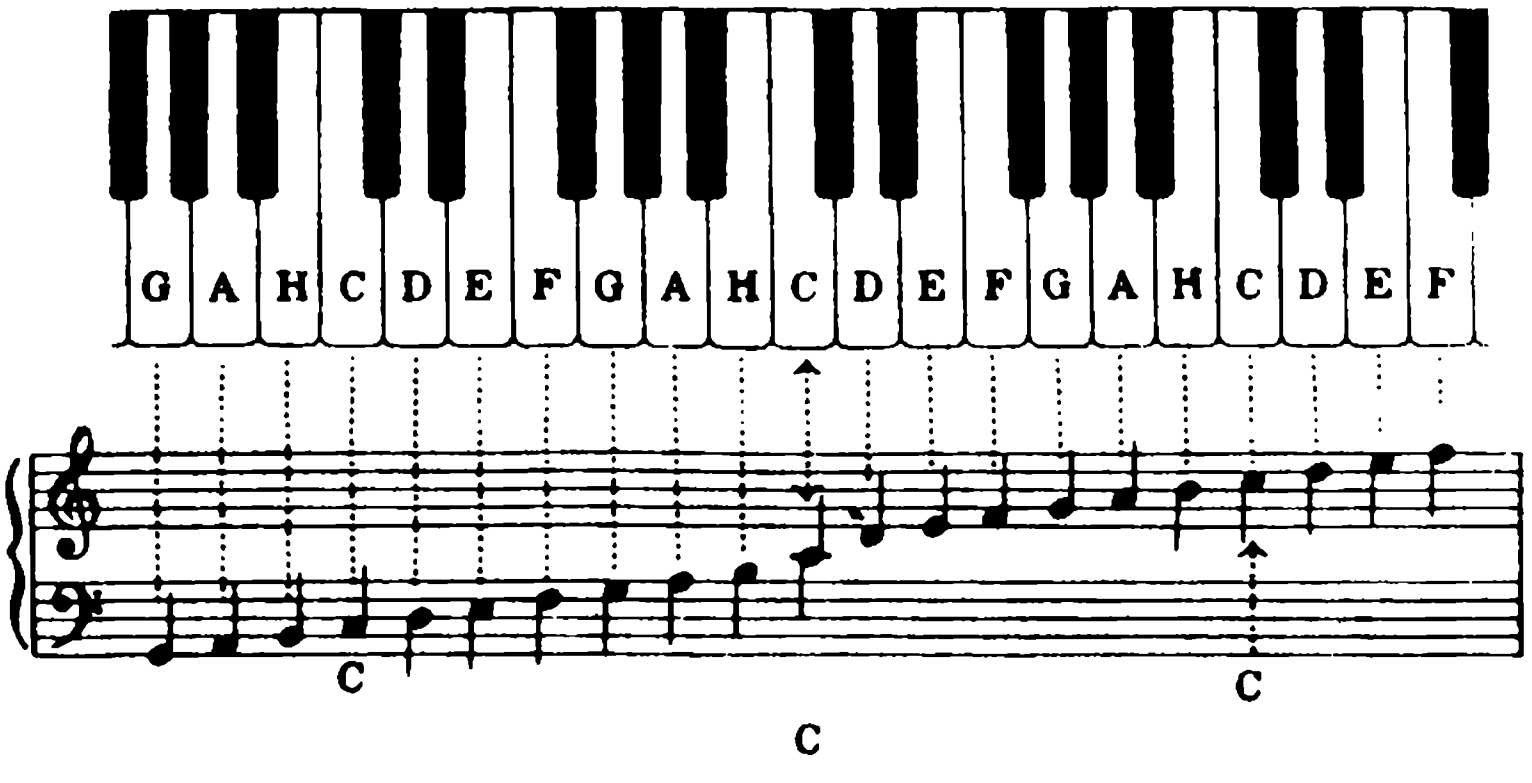
Második szólam – Hasonló az első szólamhoz, de nemcsak zenei hangokat, hanem hanghatásokhoz zajokat is tud kelteni. Írjunk kettest a parancsba zenei hangok keltésére, illetve hármast zajok, hanghatások (például mennydörgés és eső hangjának) keltésére!

A SOUND szó utáni második szám a hangmagasság (frekvencia). Ez bármilyen szám lehet 0-tól 1015-ig. Ez utasítja a Commodore 16-ot, hogy milyen alacsony vagy magas hangot játsszon. Ahogy a szám növekszik, úgy válik a hang egyre magasabbá. A legnagyobb értékek (1015 közelében) az emberi fül számára már nem hallhatók.

MEGJEGYZÉS: Ha a szólamszám 3-as, a keltett zaj a 600-tól a 940 Hz-ig terjedő tartományba eső "fehér" zaj. A tartományon kívül eső regiszterértékekkel érdekes hanghatásokat érhetünk el.

A következőben a skálán lévő hangokat szemléltetjük a hozzájuk tartozó hangmagasságnak megfelelő értékkel együtt. A Commodore 16 teljes hangjegy táblázata a függelékben található.

HANGJEGY	A	H	C	D	E	F	G
ÉRTÉK	772	800	812	836	856	865	882
TÉNYLEGES FREKVENCIA (Hz)	443.8	499.3	527.5	594.9	665.7	703.4	787.6



Próbáljuk ki a következő programot:

```
NEW
10 VOL 8                          beállítja a hangerőt
20 X = 0
30 DO
40 SOUND 1,X,5                    előállítja a hangot
50 X = X + 5
60 LOOP UNTIL X = 1015
70 VOL 0                          kikapcsolja a hangerőt
80 END
```

A program elárul valamit a Commodore 16 zenei képességeiről.
A SOUND parancs utáni harmadik szám a hang időtartamát (hosszúságát) vezérli. Azt közli a számítógéppel, hogy milyen hosszan játssza a hangot. A szám bármi lehet 0-tól 65535-ig. Ez a szám egy időzítőt állít be, ami minden hatvanad másodpercben eggyel csökken. A 60-as érték így 1 másodpercig tartja fenn a hangot. Az az ökölszabály, hogy minél nagyobb a szám, annál hosszabb ideig szólal meg a hang. Ha 65535-öt használunk, a hang több mint 16 percig szól. A hang kikapcsolásához írhatunk 0-át, ami nem engedi megszólalni a hangot.

ZENEI HANGHATÁSOK

Hanghatásokat létrehozhatunk zenei hangok vagy zajok keltésével. Az egyszerű BASIC programok és hang előállítására szolgáló utasítások kombinálása szokatlan és szórakoztató hatásokat eredményezhet. Például a FOR...NEXT...STEP hurok eredményesen használható a hangeffektusok létrehozására. Az utasításhurok működése a következő: valahányszor a számítógép a FOR-t végrehajtja, a ciklusváltozót (példánkban S) módosítja. A NEXT-et végrehajtva visszaugrik a FOR utasításra. Példánkban a

FOR...NEXT hurokban negatív lépésközzel (STEP) nagy számról indulunk, melyet 25-ösével csökkentve mindaddig a hurokban maradunk, míg el nem érjük a kisebb számot.

```
10 VOL 8                                  a hangerőt nyolcra állítja
20 FOR S= 1000 TO 700 STEP - 25           hurkot hoz létre. lefelé számlálással
30 SOUND 1, S, 1
40 NEXT S
```

Írjunk RUN-t és nyomjuk meg a RETURN billentyűt, hogy halljuk a hanghatást! A kulcs a 20-as sor, mely az 1000-től 700-ig terjedő tartományban, 25-ösével csökkentve állítja elő a számokat. A 30-as sor arra utasítja a Commodore 16-ot az időtartam 1-re állításával, ami 1/60 másodpercet jelent, hogy minden hangjegyet csak egy pillanatig játsszon. Különböző számokkal és időtartam értékekkel kísérletezve nagyon érdekes hatásokat érhetünk el.

ZAJ LÉTREHOZÁSA

A SOUND parancsban a szólam paramétereként 3-as értéket választva zajt hozhatunk létre. Ezzel különböző hanghatásokat állíthatunk elő. A következő program a 3-as szólamot használja egy szélvihar hangjainak utánzásához.

```
10 VOL 2                                  beállítja a hangerőt
20 R= INT/RND/0/*10/+ 1                   1 és 10 között véletlen számot állít elő
30 FOR X= 1 TO R
40 SOUND 3, 600+ 30* X, 10
50 NEXT X
60 FOR X= R TO 1 STEP - 1
70 SOUND 3, 600+  30*X, 10
80 NEXT X
90 T= INT/RND/0/*100/+ 30
100  SOUND 3, 600, T
110 GOTO 20
```

A 30-as és 60-as sor FOR...NEXT hurokban állítja elő a hangmagasság (frekvencia) értékeit, eggyel növelve, illetve eggyel csökkentve a 20-as sorban előállított véletlen számon alapuló értékeket. Fontos a hangmagasság változtatása, mivel a szélvihar különböző erejű szélrohamokból áll. A 40-es és a 70-es sor tartalmazza a SOUND utasításokat, melyek a zajt állítják elő. A 90-es és 100-as sor véletlenszerű késleltetést állít be a szélvihar egyenetlen jellegének utánzásához, a fütyülések időtartamának változtatásával. A program véletlen számot generál, amit egy SOUND utasítás idő-

tartamaként használunk. Ez a SOUND utasítás állandó hangmagasságú háttérzajt ad, ami a szélrohamok ellenpontját szolgáltatja. Ez igen bonyolultnak tűnik, mintha gyakorlott programozónak kellene lenni ahhoz, hogy ezt előállítsuk. Valójában ez csak kísérletezés, a keletkezett zaj sok próbálkozás eredménye. Hanghatások létrehozása zaj generálásával érdekes és nehéz feladat, mivel meg kell találni a kívánt hanghatást eredményező paramétereket. Ez csak akkor sikerülhet, ha hajlandók vagyunk a kísérletezésre.

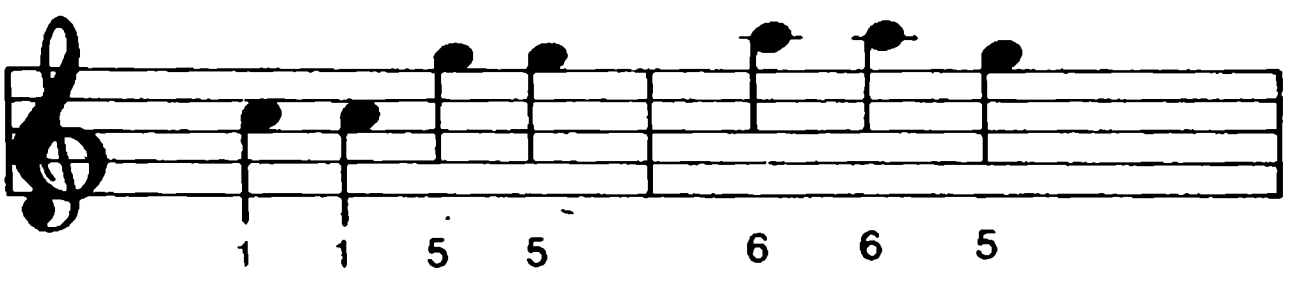
ZENE MEGSZÓLALTATÁSA

Lehet, hogy nem lesz minden érthető a következő programokban, de írjuk be őket és figyeljük meg az eredményt. A következő program zongorát szimulál, és ehhez 1-től 8-ig a számbillentyűket használja.

```
5 SCNCLR
10 FOR X= 1 TO 8: READ N(X): NEXT X
20 VOL 8
30 DO                                   nincs szóköz
40 GET A$: IF A$= "" THEN 40
50 A= ASC(A$): IF A <49 OR A >56 THEN 90
60 N= A- 48
70 SOUND 1, N(N), 5
80 COLOR 4, N, 3
90 LOOP UNTIL A= 32
100 VOL 0: COLOR 4, 2, 7
110 DATA 169, 262, 345, 383, 453, 516, 571, 596
```

A hangok megszólaltatásához üssük le 1-től 8-ig a számbillentyűket!
A képernyő kerete is színt vált a különböző hangok leütésekor. Ha a játékot befejeztük, nyomjuk meg a szóköz billentyűt a program leállításához.
Miután a Commodore 16-ot zongoraként tudjuk használni, sürgető kényszert érzünk, hogy egy népszerű dalt játsszunk. A következő számokat kell leütni a klasszikusnak tekinthető jól ismert dal lejátszásához.

```
1 1 5 5 6 6 5
4 4 3 3 2 2 1
5 5 4 4 3 3 2
5 5 4 4 3 3 2
1 1 5 5 6 6 5
4 4 3 3 2 2 1
```



A következő program DATA utasítások elemeit olvasva szólaltatja meg a dalt. A számítógép a DATA utasításban szereplő számokat változók (X és Y)

értékeként olvassa, és minden hurokban megváltoztatja azok értékét. A **DATA** utasításban szereplő számok páronként összetartoznak. Az első szám a **SOUND** utasításban a hangmagasságot, a második pedig az időtartamot határozza meg.

Evezős csónak

```
10 VOL 8
20 DO
30 READ X, Y
40 SOUND 1, X, Y
45 FOR D= 1 TO Y*16+ 30:NEXT D
50 LOOP UNTIL X= 0
60 END
```

Ez a hurok rövid késleltetést eredményez a hangok között

```
100 DATA 169, 45, 169, 45, 169, 30
110 DATA 262, 15, 345, 45, 345, 30
120 DATA 262, 15, 345, 30, 383, 15
130 DATA 453, 60, 596, 45, 453, 45
140 DATA 345, 45, 169, 45, 453, 30
150 DATA 383, 15, 345, 30, 262, 15
160 DATA 169, 60
200 DATA 0,0
```

A NAGY COMMODORE 16 ZENEGÉP

Az utolsó program valamivel hosszabb. Ez a "NAGY COMMODORE 16 ZENEGÉP". Ha megnyomjuk 1-től 9-ig a billentyűket, megszólal a hang, és a hangjegy megjelenik a kottán a megfelelő vonalon.

```
5 GOSUB 1000
6 FOR X= 1 TO 9: READ N(X): NEXT X
8 CHAR 1, 8, 1,"*A NAGY ZENEGÉP*"
10 VOL 7
20 DO
30 GET A$: IF A$ = "" THEN 30
35 A= ASC(A$): IF A <49 OR A >57 THEN 50
36 N= A - 48
40 SOUND 1, N(N), 4
45 GSHAPE N$, 150, 8 * (6+ (9- N)), 4
46 FOR Z= 1 TO 50: NEXT Z
47 GSHAPE N$, 150, 8 * (6+ (9- N)), 4
50 LOOP UNTIL A= 32
55 VOL 0: GRAPHIC 0: SCNCLR
```

```
60 END
100 DATA 345, 383, 453, 516, 571, 596, 643, 685, 704
1000 GRAPHIC 1,1
1010 FOR Y= 60 TO 124 STEP 16
1020 DRAW 1, 100, Y TO 200, Y
1030 NEXT Y
1040 A$= "FEDCHAGFE"
1050 FOR X= 1 TO 9: C= 13
1060 IF INT(X/2)= X/2 THEN C= 14
1070 CHAR 1, C, X+ 6 MID$ (A$, X, 1), 0
1075 CHAR 1, C + 10, X+ 6, RIGHT$ (STR$ (10– X), 1)
1080 NEXT X
1090 FOR X= 1 TO 8: FOR Y= 11 TO 16: DRAW  1, X, Y: NEXT Y, X
1100 Y= 1: X= 8: DRAW 1, 8, 16 TO X, Y
1110 SSHAPE N$, 1, 1, 8, 16
1120 GSHAPE N$, 1, 1, 4
1130 RETURN
```

Mint látjuk a zene és a hang megszólaltatása használható programok kiemelésére, vagy lehet a program célja is. Az ebben a fejezetben található példák ízelítőt adnak a Commodore 16 zenei képességeiről. Ne féljünk új hangok és zajok kipróbálásától, és ne habozzunk saját mesterművünket megalkotni.

Eddig megismerkedtünk néhány dologgal, amit a Commodore 16 számítógépen végrehajthatunk. A kézikönyvnek az a célja, hogy ízelítőt adjon a számítógépről, hogy új dolgok kipróbálására serkentsen, hogy a számítógép használatára tanítson és élvezetet nyújtson a futások során. De még annyi minden van... például a programozás. A kézikönyv már érintette, hogyan kell elkezdeni a BASIC használatát, de ez semmi esetre sem oktató vagy BASIC kézikönyv. A BASIC ismerettár, a következő fejezet a Commodore 16 ÖSSZES BASIC utasításának teljes listáját adja magyarázatokkal és példákkal. Előfordulhat, hogy az utolsó fejezetek példáiban nem volt érthető, hogy az egyes utasítások pontosan mit csináltak. A BASIC nyelvű programozásról úgy tanulhatunk többet, hogy olyan könyveket olvasunk, melyek a BASIC nyelvet tanítják. Az ilyen könyvek listája a függelékben található. Ha igazán érdeklődnek a programozás iránt, a Programozói Hivatkozási Kézikönyvet ajánljuk, mely rávilágít a programozás hogyanjára és miértjére, és fölfedi a titkos és nem-annyira-titkos trükköket.

BASIC 3.5

Ismerettár

- ◉ Bevezetés
- ◉ Parancsok
- ◉ Utasítások
- ◉ Függvények
- ◉ Változók és műveleti jelek
- ◉ Rövidítési és hivatkozási táblázat

BEVEZETÉS

Ebben a kézikönyvben láttunk már számos példát a BASIC nyelv használatára, benyomást szerezhettünk a számítógép-programozásról és megismertünk néhány kulcsszót. Az ismerettár a BASIC 3.5 nyelv előírásainak és terminológiájának (szintaxisának) egészét felöleli, az egyes utasítások tömör leírásával. Nyugodtan kísérletezhetünk az utasításokkal, hiszen programmal nem lehet tönkre tenni a Commodore 16-ot, és ne feledjük, a tanulás legjobb módja maga a programozás.

Az ismerettár tartalmazza a BASIC 3.5 parancsainak és utasításainak formátumát, rövid leírását, és példákat is közöl. A könyvet nem BASIC tankönyvnek szánjuk. Aki a BASIC-et tanulni akarja, annak a G függelékben található tankönyveket ajánljuk.

A parancsokat és utasításokat különálló részben ismertetjük. Az egyes részeken belül a parancsokat és utasításokat ábécé sorrendben tárgyaljuk. A parancsokat általában közvetlen üzemmódban (parancsmódban) használjuk míg az utasításokat leggyakrabban programokban. A legtöbb esetben a parancsok utasításokként is használhatók a programokban, ha sorszámmal látjuk el őket. Ugyanakkor sok utasítást parancsként is használhatunk közvetlen üzemmódban (azaz sorszámok nélkül). Ha bizonytalanok vagyunk abban, hogy egy kulcsszó utasítás vagy parancs-e, a hivatkozási táblázatban keressük először.

A BASIC ismerettár szervezése a következő:

PARANCSOK: a parancsokat programokkal kapcsolatban szerkesztésre, tárolásra és törlésre használjuk.

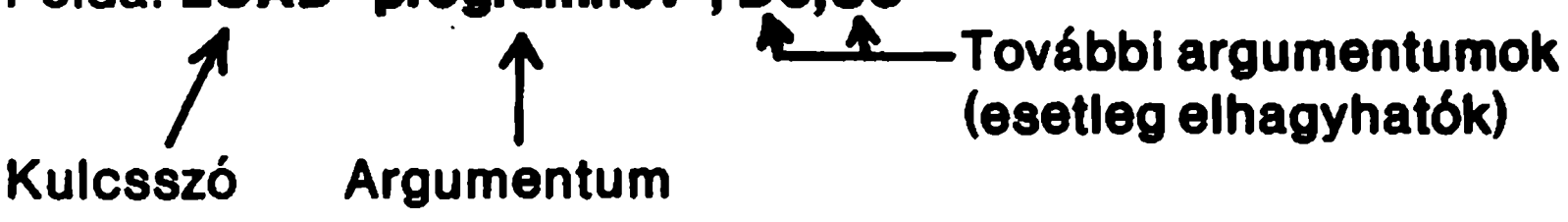
UTASÍTÁSOK: a BASIC nyelv utasításait számozott programsorokban alkalmazzuk.

FÜGGVÉNYEK: karaktersorozatot kezelő, numerikus és nyomtatási függvények.

VÁLTOZÓK ÉS MŰVELETI JELEK: különböző típusú változok, érvényes változónevek, aritmetikai és logikai műveleti jelek.

A PARANCSOK ÉS UTASÍTÁSOK FORMÁTUMA

Az ezen fejezetben bemutatott parancsok és utasítások leírása következetes jelölésmódot alkalmaz, hogy a lehető legérthetőbbé tegye alkalmazásukat. A legtöbb esetben számos példa illusztrálja a szóban forgó parancsot. A következő példa a BASIC parancsok és utasítások leírásakor alkalmazott jelölésmódot szemlélteti:



Az utasítás vagy parancs azon részét, melyet pontosan úgy kell beírni, ahogy megjelenik – mint pl. a parancs nevét – , nagy betűkkel írjuk. Azokat a szavakat, melyeket nem pontosan ugyanúgy kell beírni – pl. a program nevét – , kis betűvel szedték. Ha idézőjelek szerepelnek (általában program vagy adatállomány nevét keretezően), a parancsnak vagy utasításnak ugyanúgy kell ezeket tartalmaznia, mint a formátum listán.

- A KULCSSZAVAK nagybetűvel írva vastagbetűs szedéssel jelennek meg. EZEKET A KULCSSZAVAKAT VÁLTOZTATÁS NÉLKÜL KELL BEÍRNUNK. Ámbár sok kulcsszónak van rövidítése is, melyet szintén használhatunk (lásd a hivatkozási táblázatot).

A kulcsszavak a BASIC nyelv azon részei, melyeket a számítógép fölismer. A kulcsszó a parancs vagy utasítás központi része. Ezek közlik a számítógéppel, hogy mit kell végrehajtania. Ezeket a szavakat nem használhatjuk változók neveként.

- Az argumentumok (vagy paraméterek) kisbetűvel írva jelennek meg. Az argumentumok a parancsok és utasítások olyan részei, amelyeket mi választunk; ezek egészítik ki a kulcsszavakat a parancsok vagy utasítások végrehajtásához szükséges egyedi információval. Egy kulcsszó például egy program betöltésére utasítja a számítógépet, míg egy argumentum a betöltendő program nevét adja meg, egy további argumentum pedig a lemezmeghajtó címét közli, amelyikben a programot tartalmazó lemez található. Az argumentumok lehetnek állománynevek, változók, sorszámok, stb.
- A SZÖGLETES ZÁRÓJELEK [] VÁLASZTHATÓ, opcionális argumentumokat jelölnek. A felsorolt argumentumok bármelyike választható vagy elhagyható saját igényeinktől függően.
- CSÚCSOS ZÁRÓJEL <> jelöli, hogy felsorolt argumentumok közül egyet választanunk KELL.
- FÜGGŐLEGES VONAL | választja el az argumentum lista olyan elemeit, melyekből választani kell, és más elemek nem választhatók. Ha a függőleges vonal SZÖGLETES ZÁRÓJELBE zárt elemek között jelenik meg, választásunk a listán lévő elemekre korlátozódik, de lehetőségünk van a paraméter elhagyására. is.
- SZÓKIHAGYÁS... három pont egymás után, azt jelenti, hogy egy opció vagy argumentum egynél többször ismételhető.
- IDÉZŐJELEK "" karaktersorozatokat, állományneveket és egyéb kifejezéseket határolnak. Ha egy argumentum idézőjelek között szerepel, a

parancsban vagy utasításban is meg kell tartani az idézőjeleket. Az Idézőjelek a parancs vagy utasítás el nem hagyható részei.

- ◉ GÖMBÖLYŰ ZÁRÓJELEK: ha az argumentumok gömbölyű zárójelben vannak, a parancsban vagy utasításban is szerepelnie kell ezeknek. A gömbölyű zárójelek sem hagyhatók el, ha parancs vagy utasítás leírásában szerepelnek.
- ◉ A VÁLTOZÓ bármely érvényes BASIC változó nevére vonatkozik, például X, A$, vagy T%.
- ◉ A KIFEJEZÉS bármilyen BASIC kifejezést jelent, például A+ B+ 2, vagy .5* (X+ 3).

BASIC PARANCSOK

AUTO

AUTO [növekmény]

Elindítja az automatikus sorszámozást, ami megkönnyíti a programok beírását, mivel nem kell törődni a sorszámokkal. A programsorok beírása után a RETURN hatására a következő sorszám megjelenik a képernyőn, a kurzor pedig az új sor kezdetét jelzi. A [növekmény] argumentum a sorok közötti növekményt jelenti. Ha az AUTO parancs argumentum nélkül áll, megszünteti az automatikus sorszámozást. Ugyanez a RUN hatása is. Ez a parancs csak direkt módban hajtható végre.

Példák:

AUTO 10	Automatikusan 10-es növekménnyel számozza a sorokat
AUTO 50	Automatikusan 50-es növekménnyel számozza a sorokat
AUTO	Megszünteti az automatikus sorszámozást

BACKUP

BACKUP Dmeghajtó# TO Dmeghajtó# [,QN Uegység#]

Ez a parancs átmásolja egy lemez minden állományát egy másik lemezre kettős lemezmeghajtó esetén. Új lemezre is másolhatunk anélkül, hogy először a HEADER parancsot használnánk az új lemez inicializálására, mert a BACKUP parancs minden információt átmásol a lemezre, így a formátumot is. Fontos lemezeket ajánlatos duplikálni az eredeti elvesztése vagy megsérülése esetére.

Mivel a BACKUP parancs inicializálja is a lemezeket, le is töröl mindent, ami korábban azon a lemezen volt, amire másolunk. Ezért, ha egy előzőleg használt lemezre másolunk, győződjünk meg róla, hogy nem tartalmaz olyan programot, amire még szükségünk van. Lásd még a COPY parancsot!

MEGJEGYZÉS: Ez a parancs csak kettős lemezmeghajtó esetén használható.

Példák:

BACKUP D0 TO D1	Minden állományt átmásol a 0-ás meghajtóban lévő lemezről az 1-es meghajtóban lévő lemezre.
BACKUP D0 TO D1 ON U9	Minden állományt átmásol a 0-ás meghajtóról az 1-es meghajtóra a 9-es című egységen.

COLLECT

COLLECT [Dmeghajtó#][,ON Uegység#]

Ez a parancs felszabadítja a le nem zárt állományok által elfoglalt helyet, és törli az ezen állományokra való hivatkozást a tartalomjegyzékből.
Példa:
COLLECT D0

CONT (CONTINUE)

Ez a parancs arra használható, hogy újraindítsunk egy a STOP billentyű, a STOP utasítás vagy a programban szereplő END valamelyikével megállított programot. A program végrehajtása onnan folytatódik, ahol megszakítottuk. A CONT nem működik, ha a program sorait megváltoztattuk, vagy kibővítettük (illetve akkor sem, ha a kurzort egy programsorra mozgattuk, és megnyomtuk a RETURN-t anélkül, hogy bármit is megváltoztattunk volna), vagy ha a program hiba következtében állt le, illetve hibát okoztunk a program újraindítása előtt. Ebben az esetben a hibaüzenet a következő: CAN'T CONTINUE ERROR (hiba miatt nem folytatható).

COPY

COPY [Dmeghajtó#,] "forrásállomány" TO [Dmeghajtó#] "másik állomány" [,ON Uegység#]

Az egyik meghajtóegységben lévő lemez egyik állományát (a forrásállományt) átmásolja a kettős meghajtó egységben lévő másik lemezre, vagy másolatot készít az állományról ugyanazon a meghajtón (eltérő állománynévvel).

Példák:

COPY D0,"DÉL" TO D1,"ÉJSZAKA"	A Delet a 0-ás meghajtóról átmásolja az 1-es meghajtóra, Éjszaka néven
COPY D0,"VACAK" TO D1,"VACAK"	Átmásolja a Vacak nevű állományt a 0-ás meghajtóról az 1-esre
COPY D0 TO D1	Minden állományt átmásol a 0-ás meghajtóról az 1-esre
COPY "MACSKÁK" TO "KUTYÁK"	A Macskák nevű programról másolatot készít Kutyák névvel ugyanazon a meghajtóegységen.

DELETE

DELETE [első sorszám] [-utolsó sorszám]

Törli a BASIC program sorait. Ez a parancs csak parancsmódban hajtható végre.

Példák:

DELETE 75	Törli a 75-ös sort.
DELETE 10-50	Törli a sorokat 10-től 50-ig, beleértve a megadottakat.
DELETE -50	Törli a programsorokat a program kezdetétől az 50-es sorig bezárólag.
DELETE 75-	Törli az összes sort a 75-östől a program végéig.

DIRECTORY

DIRECTORY [Dmeghajtó#] [,Uegység#] [,"állománynév"]

A lemez tartalomjegyzékét jeleníti meg a Commodore 16 képernyőjén. Használjuk a CTRL-S billentyűket a kijelzés megállítására (bármely egyéb

billentyű újra indítja a listázást). A C= billentyű (a Commodore billentyű) lelassítja a kijelzést. A DIRECTORY parancs nem használható a lista kinyomtatására. Ehhez be kell töltenünk a lemez tartalomjegyzékét. (Ezzel felülírva az éppen a memóriában lévő programot.)
Példák:

DIRECTORY	A lemezen lévő összes állományt kiírja.
DIRECTORY D1,U9,"MUNKA"	A 9-es meghajtó egységben lévő lemez (8 az alapértelmezés) 1-es meghajtóján lévő MUNKA nevű állományt kilistázza.
DIRECTORY "AB*"	Kilistáz minden állomány, mely az AB betűkkel kezdődik, például Ablak, Abrosz, stb.
DIRECTORY D0,"ADAT ?.BANK"	A ? helyén bármely karakter lehet az adott pozición: ADAT 1.BANK ADAT 2.BANK ADAT 3.BANK mind megfelelőek.

MEGJEGYZÉS: A 8-as egységben lévő 0-ás meghajtó tartalomjegyzékének kinyomtatásához a következőképpen járjunk el:

```
LOAD "$0",8
OPEN4,4:CMD4:LIST
PRINT#4: CLOSE 4
```

DLOAD

DLOAD "állománynév" [,Dmeghajtó#] [,Uegység#]

Ez a parancs lemezről programot tölt a memóriába. (Kazettáról való betöltéskor használjuk a LOAD parancsot.) Meg kell adnunk a program nevét.
Példák:

DLOAD "KOCSI"	Megkeresi és betölti a lemezen levő Kocsi nevű programot.
DLOAD (A$)	Betölt egy programot a lemezről, melynek neve az A$ változóban található. Hibajelzést kapunk, ha az A$ üres.

A DLOAD parancs BASIC programon belül is használható egy lemezen lévő másik program betöltésére és elindítására. Ezt nevezzük láncolásnak.

DSAVE

DSAVE "állománynév" [,Dmeghajtó#] [,Uegység#]

Ez a parancs programot tárol lemezen. (Kazettán való tároláshoz használjuk a SAVE parancsot.) A programnevet meg kell adni.
Példák:

DSAVE "DNAP" — Lemezen megőrzi a Dnap nevű program.

DSAVE (A$) — Tárolja a lemezen azt a programot, melynek neve az A$ változóban található.

DSAVE "PROG 3",D0,U9 — Tárolja a Prog 3 nevű programot a 9-es számú egység 0-ás meghajtóján.

HEADER

HEADER "lemeznév",Dmeghajtó#[,Ii.d#] [,ON Uegység#]

Minden mágneslemezt az első használat előtt a HEADER paranccsal inicializálni kell. Ha egy teljes lemezt törölni szeretnénk újbóli használat céljából, akkor is használhatjuk a HEADER parancsot. Ez a parancs a lemezt blokkoknak nevezett részekre osztja, és tartalomjegyzéket készít, melyet Directorynak vagy katalógusnak nevezünk. A lemez neve bármilyen név lehet 16 karakter hosszúságig. Az i.d. szám két karakteres azonosító. Adjunk minden lemeznek egyedi i.d. számot! (Azonosító szám.) Óvatosan járjunk el a lemezek inicializálásakor, mivel a HEADER parancs minden tárolt adatot töröl. Ha nem adunk meg azonosító számot, gyors inicializálást végezhetünk. Ekkor a régi azonosítót használja a gép. A gyors inicializáló módszert csak akkor használhatjuk, ha a lemez előzőleg formázva volt, mivel a gyors inicializálás csak a tartalomjegyzéket törli, a lemez formázása nélkül.
Példák:

HEADER "LEMEZEM", I23,D0
HEADER "LABDA",I45,US

HELP

A HELP parancs akkor használatos, ha programunkban hiba van. Ha a **HELP parancsot** beírjuk, megjelenik a hibás sor, és a hibát tartalmazó rész villog.

KEY

KEY [billentyű#, karaktersor]

A Commodore 16 számítógépen 8 funkció billentyű alkalmazható: 4 alsó állású, és 4 felső állású (SHIFT).
A Commodore 16 lehetővé teszi az egyes funkció billentyűk különböző célokra történő felhasználását. A KEY utasítás paraméter megadása nélkül kilistázza az összes funkció billentyű pillanatnyi megfeleltetését. Az illető funkció billentyűhöz rendelt adat megjelenik a képernyőn, ha a billentyűt lenyomjuk. A definiciók teljes hossza maximum 128 karakter lehet együttesen. Teljes parancsok, vagy parancsok sorozata feleltethető meg egy funkció billentyűnek.
Például:
KEY 7,"GRAPHIC0"+ CHR$(13)+ "LIST"+ CHR$(13)
hatására a számítógépet szöveg üzemmódba helyezzük, és kilistázzuk a programot, az F7 billentyű megnyomásával (direkt módban). A CHR$(13) a RETURN karakter ASCII kódja. Karaktersorozaton belül a CHR$(34) kódot kell használni idézőjelek helyett.
A funkció billentyűk programban újra definiálhatók.
Például:

10 KEY2,"TESTING"+ CHR$(34): KEY3,"NO"

10FORI= 1 TO 8:KEYI,CHR$(I+ 132): NEXT

Úgy definiálja a funkció billentyűket, ahogy azok a Commodore 64 és VIC 20 számítógépen vannak definiálva.

A funkció billentyűk alapértelmezés szerinti visszaállítását a Commodore 16 ki- és bekapcsolásával vagy a RESET megnyomásával érhetjük el.

LIST

LIST [első sor]-[- utolsó sor]

A LIST parancs lehetővé teszi a beírt, vagy a Commodore 16 memóriájába betöltött BASIC program kiíratását. Ha a LIST parancsot sorszámok nélkül használjuk, a teljes lista megjelenik a képernyőn. A kiíratás lassítható a C= gomb megnyomásával, megállítható a CTRL-S billentyűkkel (újraindítható bármelyik billentyűvel), illetve végleg leállítható a RUN/STOP billentyű segítségével. Ha a LIST szót sorszám követi, a Commodore 16 csak azt a sort listázza ki. Ha a LIST szót két szám követi, melyek között kötőjel van, a Commodore 16 kilistáz minden sort, amelyik az első sorszámtól a másodikig

terjed. Ha a LIST parancsot úgy írjuk be, hogy egy szám és egy kötőjel követi, az összes sor megjelenik a sorszámtól a program végéig. Végül ha beírjuk a LIST parancsot, egy kötőjelet és utána egy számot, megjelenik az összes sor a program elejétől az adott sorszámig. Ezen változatok használatával a progam bármely részét ellenőrizhetjük, vagy módosítás céljából megjeleníthetjük a képernyőn.

Példák:

LIST	Kilistázza az egész programot.
LIST 100-	A 100-as sortól a program végéig listáz.
LIST 10	Csak a 10-es sort listázza.
LIST -100	A program kezdetétől a 100-as sorig listáz.
LIST 10-200	A 10-estől a 200-as sorig listáz, ezeket is beleértve.

LOAD

LOAD ["programnév"[,eszköz#] [,áthelyezés jelzése]]

Ez a parancs használandó, ha kazettán vagy lemezen tárol programot akarunk betölteni. Ha csupán a LOAD-ot írjuk be és megnyomjuk a RETURN-t, a Commodore 16 képernyője törlődik. Nyomjuk meg a PLAY gombot és a Commodore 16 programot kezd keresni a mágnesszalagon. Ha talál egyet, a Commodore kiírja: FOUND "programnév". A C= billentyűvel betölthetjük a megtalált programot. Ha nem ütünk le billentyűt a számítógép kis idő múlva folytatja a keresést. Ha a program betöltése befejeződött futtathatjuk, listázhatjuk, vagy módosíthatjuk.

Beírhatjuk a LOAD szót úgy is, hogy egy programnév követi mely leggyakrabban egy idézőjelben lévő név ("programnév"). A nevet vessző követheti (az idézőjeleken kívül) és egy sorszám (vagy numerikus változó), mely az eszköz címét adja, ahonnan a programot betöltjük (lemez vagy szalag). Ha számot nem adunk meg, a számítógép az eszközszámot 1-nek veszi, ami a kazettás egység címe.

A LOAD paranccsal együtt használt másik eszköz általában a lemezmeghajtó, melynek címe 8.

Példák:

LOAD	Beolvassa a szalagon lévő következő progamot.
LOAD "ALAPOK"	A mágnesszalagon megkeresi az Alapok nevű programot, majd betölti, ha megtalálta.

**LOAD A$** — Azt a programot tölti be, melynek neve az A$ változóban található.

LOAD"HIDAK",8 — Megkeresi majd betölti a mágneslemezen található Hidak nevű programot.

A LOAD parancs BASIC programokon belül is használható a szalagon lévő következő program megkeresésére és elindítására. Ezt nevezzük láncolásnak.
Az áthelyezés jelzése azt határozza meg, hová töltődik a memóriában a program. 0 hatására a Commodore 16 a programot a BASIC programterület kezdetére tölti, míg 1 esetén arra a címre, ahonnan kimentettük. Az áthelyezés jelzése alapértelmezés szerint 0. Használatára általában gépi nyelvű programok betöltésekor kerül sor.

NEW

Ez a parancs törli a memóriában lévő teljes programot, és minden változót.
Legyünk körültekintőek, mikor ezt a parancsot használjuk!
A NEW parancs utasításként is használható BASIC programokban Amikor a Commodore 16 ehhez a sorhoz ér, a program törlődik és minden megáll. Normális körülmények között ez nem különösen ésszerű.

RENAME

RENAME [Dmeghajtó#,] "régi név" TO "új név" [,Uegység#]

Lemezen lévő állomány átnevezésére használatos.
Példa:

RENAME D0, "AKTIVUM" TO "PASSZIVUM" — Az Aktivum nevü állomány nevét Passzivumra változtatja.

RENUMBER

RENUMBER [új kezdő sorszám [,növekmény [,régi kezdő sorszám]]]

Az új kezdő sorszám a programban lévő első sor száma az átszámozás után. Alapértelmezés szerint 10.

A régi kezdő sorszám a program azon sorszáma, ahol az átszámozás kezdődik. Így lehetővé válik, hogy a program egy részét átszámozzuk. Alapértelmezés szerint ez a program első sora.
Ez a parancs csak direkt módban használható.
Példák:

RENUMBER 20,20,1 Az első sortól kezdődően átszámozza a programot. Az első sor a 20-as sor lesz, és a többi sor huszas növekménnyel kap sorszámot.

RENUMBER,,65 A 65-ös sorral kezdődően 10-es növekménnyel számozza át a sorokat. A 65-ös sor új sorszáma 10-es lesz (feltéve, hogy nincsenek sorszámok 10 és 65 között, különben a parancs végrehajtására nem kerül sor).

RUN

RUN [sor#]

Egy programot a memóriába való beírás vagy betöltés után a RUN paranccsal indíthatunk el. A RUN a program végrehajtásának megkezdése előtt töröl minden változót. Ha a RUN parancsot nem követi szám, a számítógép a legalacsonyabb számú programsornál kezdi a végrehajtást. Ha a RUN parancs után sorszám áll, a végrehajtás ennél a sornál kezdődik. A RUN parancsot használhatjuk programon belül is.
Példák:

RUN A program végrehajtását a legkisebb sorszámú sornál kezdi.

RUN 100 A 100-as sornál kezdi a program végrehajtását.

SAVE

SAVE ["programnév" [,eszköz#[,EOT jelzése]]]

Ez a parancs az éppen a memóriában lévő programot kazettára vagy lemezre veszi fel. Ha csak a SAVE parancsot írjuk be és megnyomjuk a RETURN billentyűt, a Commodore 16 kazettára próbálja felvenni a programot. Nincs lehetőség arra, hogy ellenőrizzük, hogy az adott helyen van-e más program a szalagon, így óvatosnak kell lenni kimentéskor. Ha a SAVE parancsot idézőjelben lévő névvel együtt, vagy karaktertváltozó nevével írjuk be, a Commodore 16 a programot ezen a néven tárolja, így később sokkal könnyebb lesz azt megtalálni, és betölteni. A SAVE parancsban az egység címét a név után vesszővel elválasztva, vagy egy numerikus változó

tartalmaként kell megadni. Az 1-es egységszám a kazettás egység, a 8-as a lemez. A szalag címe után állhat egy második szám, mely 0-tól 3-ig terjedhet, vesszővel elválasztva. Ha a második szám (1), akkor a kimentett programot a LOAD utasítás a memória azon aktuális címére tölti, amelyen a SAVE utasítás előtt állt. Kettes (2) esetén szalagvég jelet (END OF TAPE) tesz ki, és három (3) esetén az (1)-es és (2)-es eljárást együttesen alkalmazza. Kettes esetén (2) ha be akarunk tölteni egy programot, és a számítógép ilyen jelet talál a betöltendő program helyett, a következő hibajelzést kapjuk: FILE NOT FOUND ERROR (Az állomány nem található).
Példák:

SAVE Szalagra menti a programot név nélkül.

SAVE"PÉNZ" Szalagra menti a programot Pénz néven.

**SAVE A$** Szalagra veszi fel a programot azon a néven, mely az A$ változóban található.

SAVE"ÖNMAGA",8 Lemezre menti a programot, Önmaga néven.

SAVE"JÁTÉK",1,2 Szalagon tárolja a programot Játék néven és egy EOT (szalagvég) jelet tesz a program végére.

SCRATCH

SCRATCH"állománynév"[,Dmeghajtó#] [,Uegység#]

Kitöröl egy állományt a mágneslemez tartalomjegyzékéből. Figyelmeztetésként még megkérdezi: Are you sure? (Biztos benne?), mielőtt a Commodore 16 végrehajtja a műveletet. Írjunk be nagy Y-t a törlés végrehajtására, illetve N-t a művelet törlésére! Ezt a parancsot akkor használjuk, ha nem kívánt állományokat akarunk törölni.
Példa:

SCRATCH"TARTALOM",D1 Törli a TARTALOM nevű állományt az 1-es meghajtóban lévő lemezen.

VERIFY

VERIFY"állománynév" [,meghajtó#] [,áthelyezés jelzése]

Ez a parancs arra utasítja a Commodore 16-ot, hogy hasonlítsa össze a szalagon vagy lemezen lévő programot a számítógép memóriájában lévő programmal. Ezzel győződhetünk meg arról, hogy a tárolni kívánt program valóban ki van mentve, és a kimentés hibátlan. Ez a parancs a szalag pozicionálására is használható, így a felírás az utolsó program végét követően folytatódik. Ehhez a szalagon lévő utolsó program nevét kell megadni a

VERIFY parancsban. Az ellenőrzés megtörténik, a programok nem lesznek azonosak (ezt vártuk). Ezután viszont a szalag azon a helyen van, ahol megkezdhetjük a program felvitelét a regisztrált programok törlésének veszélye nélkül.
Ha a VERIFY parancsot önmagában adjuk meg, a Commodore 16 összehasonlítja a szalagon lévő következő programot tekintet nélkül annak nevére. jelenleg a memóriában lévő programmal. Ha a VERIFY parancsot idézőjelekben lévő programnév vagy karakterváltozó követi, a gép megkeresi a szalagon az adott nevű programot és azután ellenőriz. Ha a VERIFY parancsot név, vessző és szám követi, a megadott számú eszközön történik az ellenőrzés (a kazettás egységre 1, lemezre 8-as a szám). Az áthelyezés jelzése a LOAD parancsban lévővel azonos.
Példák:

VERIFY	Ellenőrzi a szalagon lévő következő programot.
VERIFY"VALÓSÁG"	Megkeresi a szalagon a Valóság nevű programot, és összehasonlítja a memóriában lévővel.
VERIFY"ÉN",8,1	Megkeresi, majd ellenőrzi a lemezen az Én nevű programot.

BASIC UTASÍTÁSOK

BOX

BOX[szín#], a1,b1 [,a2,b2] [,szög] [,festés]

szín	.....	Szín forrás (0– 3); alapértelmezés 1 (előtér szín)
a1, b1	.....	Csúcs koordináta
a2, b2	.....	az a1, b1-el szemben lévő csúcs; alapértelmezés a PC (Pixel Kurzor, lásd 119. old.)
szög	.....	Elforgatás szöge az óramutató járásával egyirányban mérve fokokban; alapértelmezés 0 fok
festés	.....	Kiszínezett forma (0: kikapcsolva, 1: bekapcsolva); alapértelmezés szerint 0

Ez a parancs lehetővé teszi tetszőleges méretű téglalap rajzolását a képernyő tetszőleges helyére. Az alapértelmezés szerinti értéket kapjuk, ha a paraméterek helyett vesszőt használunk. Az elforgatás a téglalap középpontja körül történik. A Pixel Kurzor (PC)az a2,b2 pontban marad a BOX utasítás végrehajtása után.
Példák:

BOX 1,10,10,60,70 Téglalap körvonalát rajzolja meg.

BOX, 10, 10, 60, 60, 45, 1 Kitöltött, elforgatott négyzetet rajzol

BOX, 30, 90,, 45, 1 Kitöltött, elforgatott sokszöget rajzol

CHAR

CHAR [szín #],x,y, "karakterek" [, inverz kijelzés]

szín .....Szín forrás (0 – 3)
x .....Karakter oszlop (0 – 39)
y .....Karakter sor (0 – 24)
Karakterek .....A kinyomtatandó szöveg
Inverz kijelzés .....Inverz megjelenítés jelzése (0 = kikapcsolva, 1 = bekapcsolva)

Szöveget (alfanumerikus karaktereket) lehet a képernyőn egy adott helyen a CHAR parancs segítségével megjeleníteni. A karakterek a gép karaktertárolójából kerülnek a képernyőre. A kezdő pozició x és y koordinátáit és a kijelzendő szöveget meg kell adni. A szín és az inverz megjelenítés választható.
A szöveg a következő sorban folytatódik, ha nem fér el a sorban. Ha a szöveg (TEXT) módot használjuk, a CHAR utasítás által kinyomtatott szöveg ugyanúgy jelenik meg, mintha a PRINT utasítással írtuk volna ki, beleértve az inverz megjelenítést, a kurzor-vezérlést és a villogás be- és kikapcsolását, stb. Az idézőjelen belüli vezérlő funkciók hatástalanok, ha a CHAR utasítást GRAFIKUS üzemmódban alkalmazzuk.

CIRCLE

CIRCLE[sz],[a,b],xr[,[yr][,[sa][,[ea][,[szög][,növekmény]]]]]

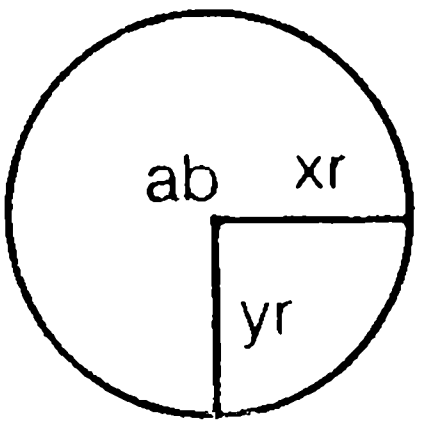
sz ..... Szín forrás 0-3
a,b ..... Középpont koordinátája (alapértelmezés a Pixel Kurzor /PC/)
xr ..... X sugár
yr ..... Y sugár (alapértelmezés az X sugár)
sa ..... Körív kezdő szöge (alapértelmezés: 0)
ea ..... Körív végszöge (alapértelmezés: 360)

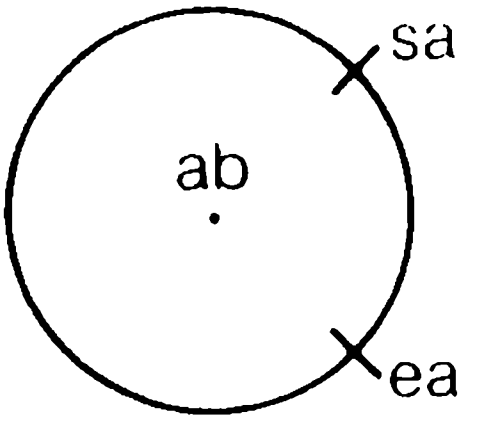
Szög ..... Forgatás az óramutató járásával megegyező irányban, fokokban (alapértelmezés: 0 fok)

Növekmény ..... A szegmensek közötti fokok száma (alapértelmezés szerint 2 fok)

A CIRCLE parancs segítségével kört, ellipszist, körívet, háromszöget, vagy sokszöget tudunk rajzolni. A végkoordináta a kör kerületén található, a körív végződésének szögénél. Minden forgatás a középpont körül történik. Ívek rajzolása a kezdő szögtől a végződés szögéig az óramutató járásával egyező irányban történik. A szegmens növekményétől függ a forma szögletessége. Kisebb növekmények lekerekítettebb formát eredményeznek.
Példák:

CIRCLE, 160,100,65,10	Ellipszist rajzol
CIRCLE, 160,100,65,50	Ellipszist rajzol
CIRCLE, 60,40,20,,,,,45	Nyolcszöget rajzol
CIRCLE, 260,40,20,40,,,,90	Rombuszt rajzol
CIRCLE, 60,140,20,,,,,120	Háromszöget rajzol





CLOSE

CLOSE állomány#

Ez az utasítás az OPEN utasítás által megnyitott állományt zárja le. A CLOSE utasítást követő szám a lezárandó állomány száma.
Példa:

Close 2 A 2-es logikai állományt zárja le.

CLR

Ez az utasítás az összes változót törli a memóriában, de a programot változatlanul hagyja. Az utasítás végrehajtása a RUN vagy NEW parancs, illetve szerkesztés esetén automatikusan történik.

CMD

CMD állomány [,írási lista]

A CMD a kivitelt, melyet általában a képernyőre küldünk, (például a PRINT utasítással vagy a LIST paranccsal, de a POKE kivételével) egy másik egységre irányítja. Ez lehet a nyomtató, illetve szalagon vagy lemezen lévő adatállomány. Ezt az eszközt vagy állományt előzőleg meg kell nyitni (OPEN). A CMD parancsot számnak, vagy az állományra vonatkozó numerikus változónak kell követnie.
Példák:

OPEN 1,4 Megnyitja a 4-es számú eszközt, ami a nyomtató.
CMD 1 Minden kivitel a nyomtatóra megy.
LIST A listázás, sőt a READY kiírása is a nyomtatóra megy, nem a képernyőre.
PRINT#1 A kivitelt visszaállítja a képernyőre.
CLOSE 1 Lezárja az állományt.

COLOR

COLOR forrás#, szín#[,fényesség#]

A színt az öt színforrás valamelyikének felelteti meg:

SZÁM	FORRÁS
0	Háttér
1	Előtér
2	Többszínű grafika 1
3	Többszínű grafika 2
4	Keret

Színeket az 1 – 16 tartományban használhatunk (1 fekete, 2 fehér, 9 narancs, stb (lsd a 62. old. táblazata). Lehetőségként a fényerő szintek 0 és 7 között állíthatók, 0 a legkisebb, 7 a legnagyobb fényerő. A fényerő alapértelmezés szerinti értéke 7. A fényességet 8 szinten állíthatjuk bármely színnél, a fekete kivételével.

DATA

DATA vesszővel elválasztott állandók

Ezt az utasítást a READ utasítással beolvasandó elemek listája követi. Az elemek lehetnek számok vagy szavak, vesszővel elválasztva. A szavakat

nem kell idézőjelek közé tenni, ha nem tartalmazzák a következő karaktereket: **szóköz**, pont vagy vessző. Ha két vessző között nem áll semmi, a READ **utasítás** 0 értékű számnak, vagy üres karaktersorozatnak olvassa. A DATA **utasítást** programba kell írni, különben nem ismeri föl a gép. Nézzük meg a RESTORE utasítást is, ami lehetővé teszi a Commodore 16 számára az **adatok** újraolvasását is.
Példa:
DATA 100,200,FRÉDI,"VILMA",,3,14,ABC123

DEF FN (Függvénydefiniálás)

DEF FN név(változó)= kifejezés

Ez az utasítás lehetővé teszi, hogy bonyolult számításokat függvényként definiáljunk. Ha egy hosszú képletet többször használunk egy programban, ezzel az utasítással sok helyet takaríthatunk meg.
A függvénynek adott név az FN betűkkel kezdődik, és bármilyen érvényes numerikus változónévvel folytatódik. Először a függvényt kell definiálni a DEF utasítás használatával, melyet a függvénynek adott név követ. A nevet gömbölyű zárójelbe () írt numerikus változó követi (esetünkben X). Ezután egyenlőségjel áll, melyet a definiálni kívánt képlet követ. A képlet által kijelölt műveleteket végrehajthatjuk, ha a változó helyére tetszés szerinti számot írunk az alábbi példa 20-as sora alapján:
Példa:

```
10 DEF FNA(X)= 12*(34.75-X/.3)+ X
20 PRINT FNA(7)
```

A DEF utasításban szereplő képletben X-et mindenhol 7 helyettesíti.

MEGJEGYZÉS: A DEF FN utasítást csak valós numerikus függvények esetén használhatjuk, egész változókkal vagy karakterváltozókkal nem.

DIM

DIM változó(indexek)[,változó(indexek)]...

Mielőtt változó tömböket használnánk, először a DIM utasítást kell végrehajtani, hogy a tömb méretét rögzítsük (kivéve, ha 11 vagy kevesebb eleme van a tömbnek). A DIM utasítást a tömb neve követi, amely bármilyen érvényes változónév lehet. Ezután gömbölyű zárójelbe téve következik egy szám (vagy numerikus változó) mely az egyes dimenziók elemeinek számát adja meg. Az egynél több dimenziós tömböt mátrixnak nevezzük. Bármennyi

dimenziót használhatunk, de vegyük figyelembe, hogy a kijelölt változók helyet foglalnak, és könnyen kifuthatunk a memóriából. A DIM utasítással létrehozott változók számának meghatározásához szorozzuk össze a tömb minden dimenziója elemeinek számát. (Minden tömb a 0-ás elemmel kezdődik.)

MEGJEGYZÉS: Egészekből álló tömbök a lebegőpontos tömbök által igényelt hely 2/5-öd részét foglalják el.

Példa:

10 DIM A$(40),B7(15), CC%(4,4,4)

41 elem 16 elem 125 elem

Egyetlen DIM utasításban egynél több tömböt is dimenzionálhatunk, ha a tömböket vesszővel választjuk el. Ha egy programban egy tömböt egynél többször dimenzionálunk, újradimenzionálási hibaüzenetet kapunk. Bevált programozói gyakorlat, hogy a DIM utasításokat a program elejére tesszük.

DO (LOOP) WHILE (UNTIL EXIT)

DO[UNTIL Boole kifejezés | WHILE Boole kifejezés]utasítások[EXIT]
LOOP[UNTIL Boole kifejezés | WHILE Boole kifejezés]

(Példa a Boole kifejezésre: A = 1 vagy H >= 57.)

Végrehajtja az utasításokat a DO és a LOOP utasítások között. Ha a DO vagy LOOP utasítást nem módosítja UNTIL, vagy WHILE, a közbeeső utasítások végrehajtása végtelen ciklusban folytatódik. Ha EXIT utasítás fordul elő a DO hurokban, a végrehajtás a LOOP utasítást követő első utasításnak adódik át. A DO hurkokat egymásba lehet ágyazni, a FOR-NEXT hurkok szabályait követve.
Ha az UNTIL paramétert használjuk, a program végrehajtása addig folytatódik, míg az argumentum logikailag nem teljesül (nem igaz). A WHILE paraméter az UNTIL paraméter ellentéte: a program addig hajtja végre a hurkot, amíg az argumentum igaz.
Példa:

```
50 DO UNTIL X= 0 OR X= 1
        :
100 LOOP

50 DO WHILE A$= "": GET A$: LOOP
```

DRAW

DRAW[szín#][,a1,b1][,TO a2,b2,][...]

Ezen paranccsal egyedi pontokat, vonalakat és idomokat rajzolhatunk. Meg kéll adnunk a szín forrást (0-3), a kezdő (a1,b1) és végpontokat (a2,b2).
Példák:

Pont: **DRAW** 1,100,50 – nincs végpont kijelölve, a2,b2 értéke alapértelmezés szerint a1,b1; pontot hoz létre

Vonalak: **DRAW** , 10,10, TO 100,60
DRAW , TO 25,30

Idom: **DRAW**, 10,10 TO 10,60 TO 100,60 TO 10,10

END

Ha a program END utasítást hajt végre, a program futása rögtön megszakad. A CONT parancsot használhatjuk a program újraindításához az END utasítást követően.

FOR...TO...STEP

FOR ciklusváltozó= induló érték TO végérték [STEP növekmény]

Ez az utasítás a NEXT utasítással együtt többször ismétlődő programrészt alkot. Akkor használjuk, ha néhány másodperces várakozást iktatunk be, ilyenkor a számítógép egy nagy számig számol, ha meg kell számolni valamit vagy valamit többször meg kell ismételni (például nyomtatást).
A ciklusváltozó az a változó, melyet növelünk vagy csökkentünk a FOR/NEXT hurokban. Az induló érték és a végérték a ciklusváltozó kezdeti és végértéke.
A FOR utasítás logikája a következő: először a ciklusváltozó a kezdeti értékre áll be. Ha a program olyan sorhoz ér, amelyben NEXT utasítás van, a gép a ciklusváltozó értékéhez hozzáadja a STEP növekményt (ez alapértelmezés szerint 1) és ellenőrzi, hogy az így kapott érték nagyobb-e a ciklusváltozó végértékénél. Ha nem nagyobb, a FOR utasítást követő sort fogja végrehajtani a gép. Ha a ciklusváltozó nagyobb, mint a végérték, a NEXT utasítást követő utasítást fogja a gép végrehajtani.
Lásd még a NEXT utasítást is!
Példa:

```
10 FOR L = 1 TO 20
```

```
20 PRINT L
30 NEXT L
40 PRINT "HUSZONEGY! L = "L
```

A program számokat ír ki egytől húszig a képernyőre, és végül a következő üzenetet: HUSZONEGY! L = 21.

A ciklus végértékét a STEP utasítás követheti, és egy másik szám vagy változó. Ebben az esetben a STEP utasítást követő érték adódik hozzá a ciklusváltozóhoz. Így lehetővé válik, hogy visszafelé számoljunk, vagy törtrésszel módosítsunk. A ciklusokat egymásba ágyazhatjuk. Ezt egymásba skatulyázásnak nevezik. Ilyenkor vigyázni kell, hogy az utoljára megnyitott hurkot zárjuk le legelőször.

Példa egymásba ágyazott hurkokra:

```
10 FOR L = 1 TO 100
20 FOR A = 5 TO 11 STEP 2
30 NEXT A
40 NEXT L
```

Ez a FOR...NEXT ciklus a nagyobb ciklusba van ágyazva

GET

GET azonosító

A GET utasítással a billentyűzetről egy karaktert olvashatunk be. A gép a leütött karaktert akkor olvassa be, mikor a GET utasítást végrehajtja. Ha nem ütünk le billentyűt, a gép üres karaktert (nem 0-át) olvas be, és a program billentyűre várás nélkül folytatódik. Nem kell a RETURN billentyűt leütni, sőt a RETURN is beolvasható a GET utasítással.

A GET utasítást azonosító, általában karakterváltozó követi. Ha numerikus változót használtunk, és nem szám billentyűt nyomtunk le, a program hibaüzenettel leáll.

A GET utasítás állhat hurokban, tetszés szerinti billentyű vagy egy folytatást eredményező billentyű leütésére várva.

A GETKEY utasítás is használható ebben az esetben. Ez az utasítás csak programon belül használható.

Példa:

```
10 GET A$:IF A$<>"A" THEN 10
```

A sor arra vár, hogy az A billentyűt megnyomjuk a folytatáshoz.

GETKEY

GETKEY azonosító

A GETKEY utasítás igen hasonlít a GET utasításhoz. A GET utasítástól eltérően a GETKEY addig vár, míg a billentyűzeten leütünk egy karaktert. Egyszerűen használható arra, hogy egy karakter beírására várjunk.
Ez az utasítás csak programon belül hajtható végre.
Példa:

10 GETKEY A$ Ez a sor egy billentyű leütésére vár. Bármilyen billentyű beírása után folytatódik a program.

GET

GET#állomány szám, azonosító

Egy előzőleg megnyitott (OPEN) eszközről vagy állományból egy karakter bevitelére szolgál. Használata megegyezik a GET utasítás használatával.
Ez az utasítás csak programon belül hajtható végre.
Példa:

10 GET#1,A$

GOSUB

GOSUB sorszám

Ez az utasítás hasonlít a GOTO utasításhoz, de itt a Commodore 16 nem felejti el, hogy honnan jött. Ha egy RETURN utasítás végrehajtására sor kerül, a program visszaugrik a GOSUB utasítást közvetlenül követő utasításra. A GOSUB utasítás célja egy szubrutin hívása. Szubrutin alkalmazása akkor célszerű, ha egy rutint a program több pontján akarunk végrehajtani. A rutin többszöri ismétlése helyett szubrutint írhatunk, amit a program különböző részeiből a GOSUB utasítással hívhatunk. Lásd még a RETURN utasítást.
Példa:

20 GOSUB 800 A 800-as sorban kezdődő szubrutint hívja és hajtja végre.

800 PRINT "SZIA": RETURN

GOTO vagy GO TO

GOTO sorszám

Egy GOTO utasítás végrehajtása után a következő végrehajtandó sort a GOTO utasításban szereplő sorszám határozza meg. Direkt módban használva, a GOTO parancs lehetővé teszi, hogy a program végrehajtását az adott sorszámnál kezdjük anélkül, hogy a változókat törölnénk.
Példa:

```
10 PRINT "ISMÉTLÉS A TUDÁS ANYJA"
20 GOTO 10
```

A 20-as sorban lévő GOTO hatására a 10-es ismétlődik, míg csak meg nem nyomjuk a RUN/STOP billentyűt.

GRAPHIC

GRAPHIC mód [,törlés]

Ez az utasítás a Commodore 16-on az 5 grafikus üzemmód egyikét állítja be:

ÜZEMMÓD	LEÍRÁS
0	Normál szöveg
1	Nagy felbontóképességű grafika
2	Nagy felbontóképességű grafika, osztott képernyő
3	Többszinű grafika
4	Többszinű grafika, osztott képernyő

Végrehajtásakor a GRAPHIC 1 – 4 egy 10K nagyságú bittérképes területet jelöl ki, és a BASIC terület a nagy felbontóképességű tárrész mögé kerül. E felosztás marad akkor is, ha a felhasználó visszatér a TEXT üzemmódhoz (GRAPHIC 0). Ha a GRAPHIC utasításban második paraméterként 1-et adunk meg, a képernyő is törlődik.
Példák:

GRAPHIC 1,1 Beállítja a nagy felbontóképességű grafikus üzemmódot, és törli a képernyőt.

GRAPHIC 4,0 Beállítja a többszínű grafikát szövegrésszel a képernyő törlése nélkül.

GRAPHIC CLR

Valójában a GRAPHIC utasítás egy változata, de a grafikus üzemmód beállítása helyett törli a grafikus üzemmód számára fönntartott 10K területet, és újra elérhetővé teszi azt a BASIC programok számára.

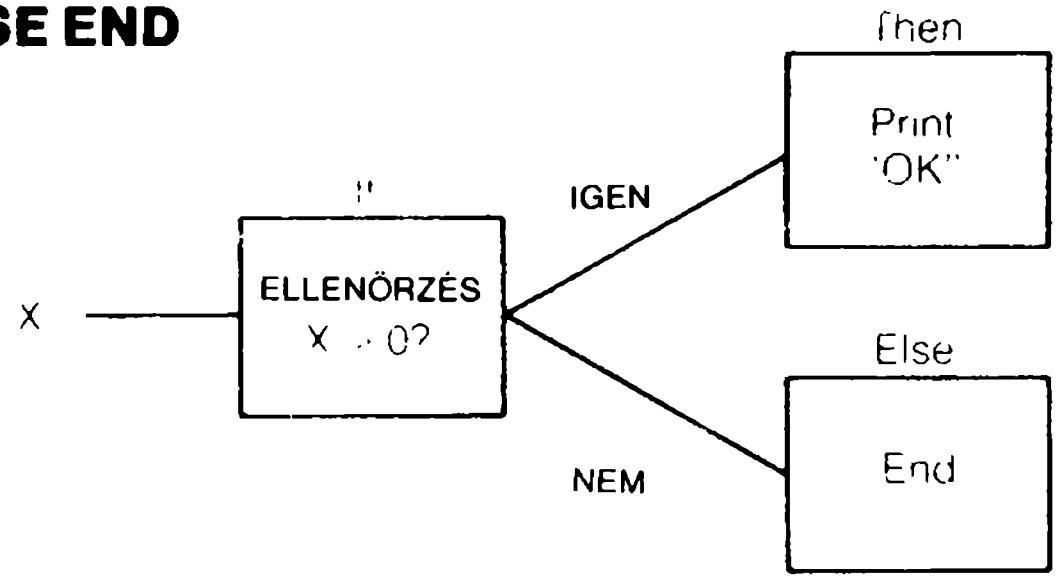
IF...THEN...:ELSE

IF kifejezés THEN kikötés [:ELSE kikötés]

Az IF...THEN utasítás egy BASIC kifejezést vizsgál, és két lehetséges változat közül az egyiket hajtja végre. Ha a kifejezés igaz, a gép a THEN utasítást követő utasítást hajtja végre. A kifejezés bármilyen BASIC utasítás lehet. Ha a kifejezés hamis, a program a következő sorra megy, ha ELSE kikötés nincs az utasításban. A kiértékelt kifejezés változó vagy művelet lehet, mely igaz, ha nem 0, és hamis, ha 0. Leggyakrabban olyan kifejezés szerepel, mely relációkat tartalmaz (= , <, >, <= , >= , <>, AND, OR, NOT).
Az ELSE utasításnak, ha szerepel, ugyanabban a sorban kell lennie, amelyikben az IF – THEN utasítások vannak. Az ELSE kikötés végrehajtására akkor kerül sor, ha a THEN kikötésére nem. Más szóval, az ELSE kikötés végrehajtására akkor kerül sor, ha az IF kifejezés hamis.
Példa:

50 IF X >0 THEN PRINT"OK":ELSE END

Ellenőrzi az X változó értékét. Ha X nagyobb mint 0, a THEN kikötést hajtja végre a gép, és az ELSE kikötést nem. Ha X kisebb mint 0, az ELSE kikötést hajtja végre a gép, a THEN kikötést pedig nem.



INPUT

INPUT ["magyarázó szöveg";]azonosító lista

Az INPUT utasítás lehetővé teszi, hogy a számítógép adatot kérjen a programot futtató személytől, és a változóba, vagy változókba helyezze az adatokat. A program leáll, kérdőjelet ír a képernyőre, és arra vár, hogy a kezelő beírja az adatokat, és megnyomja a RETURN billentyűt.

Az INPUT utasítást egy vagy több azonosító követi, vesszővel elválasztva. Szerepelhet idézőjelek között beírt üzenet is a változól: listája előtt. Ha üzenetet alkalmazunk, pontosvesszőt kell tenni a záró idézőjel után.
Ha egynél több adatot írunk be az INPUT utasítás hatására, beírás közben vesszővel kell azokat egymástól elválasztani. Ha kevesebb adatot írunk be mint ahány változó van, a számítógép a hiányzó értékekre két kérdőjellel kérdez rá. Ha a RETURN billentyűt értékek beírása nélkül nyomjuk meg, a változók megtartják előző értéküket. Ez az utasítás csak programban hajtható végre.
Példa:

```
10 INPUT "MI A NEVED";A$
20 INPUT "ÉS A KEDVENC SZÍNED";B$
30 INPUT "MEKKORA EGY FECSKE SEBESSÉGE";A
```

INPUT#

INPUT# állomány száma, azonosító lista

Ugyanúgy működik mint az INPUT, de az adatot egy előzőleg megnyitott (OPEN) állományból vagy eszközről veszi. Üzenet nem adható. Csak program módban használhatjuk.
Példa:

```
INPUT#2, A$, C, D$
```

LET

[LET] változó = kifejezés

A LET szót alig használjuk a programokban, mivel nem szükséges kitenni, de az utasítás maga a BASIC programok lényege. Amikor egy változót definiálunk; vagy értéket adunk neki, mindig a LET utasítást hajtjuk végre. A változó név, mely a számítás során értéket kap az egyenlőségjel bal oldalán van, a szám, vagy kifejezés pedig a jobb oldalon.
Példa:

```
10 LET A = 5
20 B = 6
30 C = A * B - 3
40 D$ = "SZIA"
```

A LET utasítást beleértjük (de kiírni nem szükséges) a 20-as, 30-as és 40-es sorokba.

LOCATE

LOCATE x-koordináta, y-koordináta

A LOCATE utasítás lehetővé teszi, hogy a pixel kurzort (PC) a képernyőn bárhová elhelyezzük. A PC a következő rajzolás kezdő pontjának éppen érvényes helyzete. A szokásos kurzortól eltérően nem látjuk a PC-t, de mozgathatjuk a LOCATE utasítással. Például:
LOCATE 160,100
Az utasítás a PC-t a nagy felbontóképességű képernyő közepén helyezi el. Addig semmi sem látható, míg nem rajzolunk valamit. A PC helyzetét úgy lehet meghatározni, hogy az RDOT(0) függvényt használjuk az x-koordinata, és az RDOT(1) függvényt az y-koordináta értékének megállapításához. A PC helyén érvényes szint az RDOT(2) kinyomtatásával kapjuk meg. (Minden rajzoló utasításban, ahol színválasztás lehetséges, szint nullától háromig választhatunk, ami a háttérnek, előtérnek, többszínű üzemmód 1-nek,többszínű üzemmód 2-nek felel meg.)

MONITOR

Ez az utasítás a BASIC-ből a beépített gépi nyelvű monitor programnak adja át a vezérlést. A MONITOR fejlesztésre, hibakeresésre és gépi nyelvű programok végrehajtására alkalmas.
További felvilágosítást a monitor utasításait leíró részből nyerhetünk. (A monitorból X beírásával és a RETURN megnyomásával juthatunk vissza a BASIC-be.)

NEXT

NEXT [változó,....,változó]

A NEXT utasítás a FOR utasítással együtt használatos. Ha a számítógép egy NEXT utasítást hajt végre, visszatér a megfelelő FOR utasításhoz, és megvizsgálja a ciklusváltozót (részletes leírás a FOR utasításnál található). Ha a ciklus befejeződött, a végrehajtás a NEXT utáni első utasításnál folytatódik. A NEXT utasítást azonosító vagy azonosítók vesszővel elválasztott listája követheti, de állhat önmagában is. Ha nincs azonosító, az utoljára elkezdett ciklus fejeződik be. Ha változókat adunk meg, azok balról jobbra sorrendben teljesülnek.

Példa:
10 FOR L = 1 TO 10:NEXT
20 FOR L = 1 TO 10: NEXT L
30 FOR L = 1 TO 10: FOR M = 1 TO 10:NEXT M, L

ON

ON kifejezés <GOTO | GOSUB >1.sorszám [,2.sorszám...]

Ez az utasítás a GOTO és GOSUB utasításokat az IF utasítás speciális eseteként kezeli. Az ON utasítást kifejezés követi, majd a GOTO vagy GOSUB utasítások és sorszámok listája, vesszővel elválasztva. Ha a kifejezés eredménye 1, a lista első sorszámán található utasítás végrehajtására kerül sor. Ha az eredmény 2, a második sorszámra adódik a vezérlés, és így tovább. Ha az eredmény 0, vagy nagyobb mint ahány sorszám a listán szerepel, az ON utasítást követő sorra adódik a vezérlés. Ha a szám negatív, a következő hibaüzenet jelenik meg: ILLEGAL QUANTITY ERROR (Érvénytelen mennyiség).

Példa

```
10 INPUT X:IF X 0 THEN 10
20 ON X GOTO 50, 30, 30, 70
25 PRINT "ÁTMEGY RAJTA":GOTO 10
30 PRINT "TÚL NAGY":GOTO 10
50 PRINT "TÚL KICSI":GOTO 10
70 END
```

Ha X = 1, az ON a lista első sorszámára adja a vezérlést (50).
Ha X = 2, az ON a 30-as sorra adja a vezérlést, és így tovább

OPEN

OPEN állomány#[,egység#[,másodlagos cím[,"állomány neve, típus, üzemmód"]]]

Az OPEN utasítás teszi lehetővé a Commodore 16 számára a Datassette magnetofonhoz, a mágneslemezhez, a nyomtatóhoz vagy a monitor képernyőjéhez való hozzáférést. Az OPEN utasítást a logikai állomány száma követi, melyre a többi BASIC utasítás hivatkozni fog. Ez a szám 1-től 255-ig terjedhet. Az első szám után rendszerint az eszköz címe következik. 0 a billentyűzet, 3 a képernyő, 1 a magnetofon, 4 a nyomtató és 8 általában a lemez címe. A cím számjegye elé 0-át is írhatunk (például 08-at a 8-as helyett, és ezek tetszés szerint használhatók a Commodore 16-on). Célszerű

az egység címével azonos állományszámot választani, mert így mindig tudjuk, melyik állomány melyik perifériához tartozik. Az egységcím után következhet az úgynevezett másodlagos cím. Ez a magnetofon esetén olvasáskor 0, íráskor 1, és 2 akkor, ha az írás végén szalagvég jelet írunk. Lemeznél ez a szám a csatorna számára vonatkozik, nyomtatónál pedig a másodlagos címek a nyomtató vezérlésére szolgálnak. A Commodore 16 Programozói Hivatkozási Kézikönyv, vagy az egyes perifériák felhasználói kézikönyve bővebb felvilágosítással szolgál a másodlagos címekre vonatkozóan. A harmadik szám után karaktersorozat is állhat, ami a lemez működésére vonatkozó utasításokat, vagy a szalagon, illetve lemezen tárolt állomány nevét tartalmazza. A típus és üzemmód csak lemezre vonatkozhat (állomány típusok lehetnek: prg, seq, rel, usr; üzemmódok lehetnek: írás és olvasás.)

Példák

10 OPEN 3,3	Megnyitja a képernyőt
10 OPEN 1,0	Megnyitja a billentyűzetet.
20 OPEN 1,1,0,"FEL"	Megnyitja a kazettát olvasásra, a keresett állomány neve FEL.
OPEN 4,4	Megnyit egy csatornát a printer használatához.
OPEN 15,8,15	Megnyitja a lemezen a parancs csatornát.
5 OPEN 8,8,12 "TESZT,SEQ,WRITE"	Soros lemezállományt hoz létre írásra.

Lásd még a CLOSE, CMD, GET#, INPUT#, és PRINT# utasításokat, és az ST, DS, DS$ rendszerváltozókat.

PAINT

PAINT [szín][,[a,b][,üzemmód]]

szín	(0-3); (alapértelmezés szerint 1 az előtérszín)
a.b	Kezdő koordináta (alapértelmezés szerint a PC)
üzemmód	0 = az adott területet a szín forrás által meghatározott színre festi
	1 = az adott területet a háttértől különböző forrás által meghatározott színre színezi.

A PAINT utasítás lehetővé teszi egy terület kiszínezését. A kijelölt pont körüli területet kiszínezi az azonos színű mezőig (vagy bármilyen nem háttér színig, a választott üzemmódtól függően). A PC a művelet befejezése után a kezdő pozicióban (a,b) marad.

MEGJEGYZÉS: Ha a kezdőpont színe megegyezik a megnevezett szín forrással (vagy bármely nem háttér színnel, ha az 1-es üzemmódot használjuk), változás nem történik.
Példa:

10 CIRCLE , 160,100,65	Egy kör körvonalát rajzolja meg.
20 PAINT , 160,100	Kiszínezi a kört.

POKE

POKE cím, érték

A POKE utasítás segítségével a Commodore 16 RAM (írható) memóriájának tartalmát megváltoztathatjuk, és módosíthatunk sok Commodore 16 Bementi/Kimeneti regisztert. A POKE utasítást mindig két szám vagy kifejezés követi. Az első szám a számítógép memóriáján belüli cím. Ennek értéke 0-tól 65535-ig terjedhet. A második szám a beírt érték, mely 0-tól 255-ig terjedhet, és az első szám által meghatározott cím tartalmát ezen értékre változtatja. Ezzel az utasítással bármit módosíthatunk a képernyőn, egy karakter elhelyezésétől a szín megváltoztatásáig.
Példa:

10 POKE 16000,8	A 16000-es címre 8-at helyez.
20 POKE 16*1000,27	A 16000-es címre 27-et tölt.

MEGJEGYZÉS: a POKE-hoz kapcsolódó PEEK utasítást a függvények között tárgyaljuk.

PRINT

PRINT lista

A PRINT utasítás a BASIC legfontosabb kiviteli utasítása. Bár a legtöbben a PRINT utasítást tanulják meg legelőször, nagyon sok finomság van, amit el lehet sajátítani. A PRINT utasítást az alábbi elemek követhetik bármilyen összetételben:

Karakterek idézőjelben	("szövegsorok")
Azonosítók	(A, B, A$, X$)
Függvények	(SIN(23), ABS(33))
Írásjelek	(; ,)

Az idézőjelen belüli karakterek pontosan úgy jelennek meg, ahogy az idézőjelen belül szerepelnek. A változók az általuk tartalmazott értékeket jelenítik meg a kinyomtatás során (számot vagy karaktert). A függvényeknek is az értéke kerül a képernyőre. Az írásjelek segítségével az adatokat a képernyőn áttekinthető formában helyezhetjük el. A vessző a képernyőt négy oszlopra osztja, míg a pontosvessző nem hagy üres helyet. Bármelyik jel használható az utasítás utolsó elemeként. Ennek eredményeképpen a következő PRINT utasítás úgy működik, mintha az előző PRINT utasítást folytatná.
Példa

	EREDMÉNY
PRINT"SZIA"	SZIA
A$= "BARÁTOM":PRINT "SZIA, "A$	SZIA, BARÁTOM
A= 4:B= 2:PRINT A+ B	6
J= 41:PRINT J;:PRINT J-1	41 40
C= A+ B:D= A-B:PRINT A;B;C,D	4 2 6 2

Lásd még: a POS(), SPC() és TAB() függvényeket!

PRINT

PRINT# állomány#, nyomtatási lista

Van néhány különbség ezen utasítás, és az előző PRINT utasítás között. Először is, a PRINT utasítást szám követi, mely előzőleg megnyitott adatállományra vagy egységre vonatkozik. A szám után vessző áll, majd a kinyomtatandó elemek listája. A pontosvessző a PRINT utasításhoz hasonlóan működik. A vessző a legtöbb nyomtatón 10 szóközt eredményez, de lemezen is használható az adatelemek elválasztására. (Lásd a Programozói Hivatkozási Kézikönyvet és a Lemezmeghajtó Kézikönyvet!) Egyes berendezések nem tudják végrehajtani a TAB és SPC függvények utasításait.
Példa:

10 PRINT#1, "SZIA BARÁTOM", A$,B$,

PRINT USING

PRINT[#állomány szám,] USING formátumlista; nyomtatási lista

Ezen utasítások teszik lehetővé a képernyőre, nyomtatóra vagy egyéb eszközre kírandó karakterek és numerikus adatok formátumának definiálását. A kívánt formátumot idézőjelbe kell tenni. Ez a formátumlista. Ezután pon-

tosvessző következik, majd a formátumban kinyomtatni kívánt elemek. A lista változókból vagy konkrét értékekből állhat.
Például:

```
5 X= 32: Y= 100.23:A$= "MACSKA"
10 PRINT USING "$####";13.25, X,Y
20 PRINT USING "### >#";"ABC",A$
```

Ha a programot lefuttatjuk, a 10-es sor eredménye a következő:
$13.25$32.00$***

Y értéke helyett csillagok jelennek meg, mivel Y 5 jegyű, és nem fér el a formátumban (magyarázat később)

A 20-as sor eredménye:
ABC MACSK

három szóközt hagy az ABC kiírása előtt, amint azt a formátumlistában meghatároztuk.

KARAKTER	NUMERIKUS	KARAKTER
Számjel (#)	x	x
Plusz (+)	x	
Mínusz (-)	x	
Tizedespont (.)	x	
Vessző (,)	x	
Dollár jel ($)	x	
Négy hiányjel (↑↑↑↑)	x	
Egyenlőségjel (=)		x
Nagyobb jel (>)		x

A számjel egyetlen karakternek foglal helyet a kiírási mezőben. Ha az adat több karakterből áll, mint ahány számjel van a formátum mezőben, a következő történik:
Numerikus adat esetén a teljes mezőt csillagok foglalják el. Szám nem jelenik meg.
Például:

```
10 PRINT USING "####";X
```

X következő értékeire a következő nyomtatási kép jelenik meg:

```
X= 12.34      12
X= 567.89     568
X= 123456     ****
```

Karaktersorozatok esetén a karakterek csonkulnak a mező határán. Csak annyi karakter jelenik meg, ahány számjel van a formátumlistában. A csonkítás a jobb oldalon történik.

A plusz és mínusz jelek a formátum első vagy utolsó pozícióján használhatóak, de mindkét oldalon nem. A plusz jel akkor jelenik meg, ha a szám pozitív. A mínusz jel pedig akkor, ha a szám negatív.

Ha mínusz jelet használunk, és a szám pozitív, a mínusz jelnek megfelelő pozíción szóköz jelenik meg. Ha numerikus adat esetén nem használunk se mínusz, se plusz jelet a formátum mezőben, mínusz jel jelenik meg az első számjegy vagy dollár előtt, ha a szám negatív és szóköz, ha a szám pozitív. Így tehát egy számjeggyel többet nyomtathatunk, ha a szám pozitív. Ha a# jellel és a + vagy − jelekkel kijelölt mezőbe túl sok számot akarunk írni, a mező csillagokkal töltik föl.
A tizedespont a formátumlistában a tizedespont helyét jelöli a számban. Egy formátum mezőn belül csak egy tizedespont lehet. Ha nem jelölünk ki tizedespontot a formátum mezőben, a szám a hozzá legközelebb első egész számra kerekítve tizedes értékek nélkül jelenik meg. Ha tizedespontot jelölünk ki, a tizedespontot megelőző jegyek száma (beleértve a mínusz jelet is, ha a szám negatív) nem haladhatja meg a jelek számát a tizedespont előtt. Ha a szám túl hosszú, túlcsordulás keletkezik, és a mező csillagokkal töltődik fel.
A vessző lehetővé teszi, hogy vesszőket helyezzünk el numerikus mezőkben. A formátumlistában megadott vessző azt jelöli, hogy hol jelenjen meg a vessző a kinyomtatott számban. Csak a számon belüli vesszők jelennek meg. Az első számjegytől balra álló nem használt vesszők kitöltő karakterként szerepelnek. Legalább egy jelnek meg kell előznie a mezőben az első vesszőt. Ha vesszőket jelölünk ki a mezőben, és a szám negatív, a mínusz jel jelenik meg első karakterként még akkor is, ha a karakterpozició vesszőként van kijelölve.
Példák:

MEZŐ	KIFEJEZÉS	EREDMÉNY	MAGYARÁZAT
##.#+	-.01	0.01-	A nulla egész is megjelenik.
##.#-	1	1.0	A nulla tizedes is megjelenik.
####	-100.5	-101	Egészre kerekít.
####	-1000	****	Túlcsordulás, mivel négy számjegy és a mínusz jel nem fér el a mezőben.
###.	10	10.	Tizedespontot is hozzáad.
#$##	1	$1	Dollár jelet is ír.

A formátumlistában megadott dollár jel hatására dollár jelenik meg a szám előtt. Ha azt akarjuk, hogy a dollár mindig közvetlenül a szám elé kerüljön, akkor legalább egy # jelet kell a dollár elé írni. Ha a dollárt vezető # jel nélkül adjuk meg, a dollár arra a pozícióra kerül, amelyiken a formátum mezőben van.

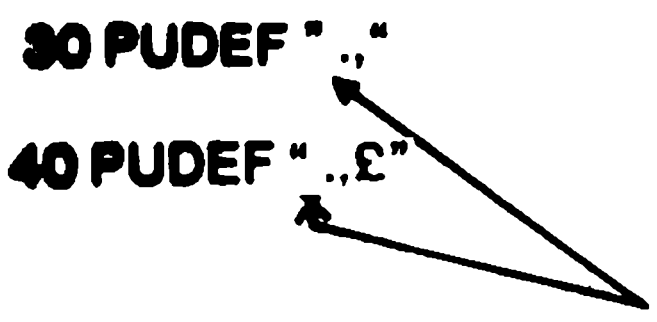
Ha a vesszőket és/vagy plusz, illetve mínusz jelet jelölünk ki a formátum mezőben a dollárral együtt, a program vesszőt vagy előjelet ír a dollár elé.
A négy felfelé mutató nyíl, vagy hiányjel segítségével egy számot tudományos jelöléssel E + formátumban tudunk kinyomtatni. # jelet kell írni a hiányjelekkel együtt a mezőszélesség kijelölésére. A hiányjeleknek a # jel után kell megjelenniük a formátum mezőben. Négy hiányjelet kell megadni, ha a számot E − formátumban akarjuk kiiratni (tudományos jelölés). Ha egynél több, de négynél kevesebb hiányjelet írunk, szintaktikai hibát kapunk. Ha négynél több hiányjelet adunk meg, csak az első négy használatára fog sor kerülni. Az ötödik és a további hiányjelek fölöslegesek.
Egyenlőségjelet használhatunk egy karaktersorozatnak a mező közepén történő elhelyezéséhez. A mező szélességét a # és = karakterek számával jelölhetjük ki a formátum mezőben. Ha a karaktersorozat rövidebb a kijelölt mezőnél, a mező közepén fog elhelyezkedni, míg ha több karaktert tartalmaz, mint amennyi a mezőbe fér, a jobb szélen levő karakterek csonkulnak, és a karatersorozat kitölti az egész mezőt.
A nagyobb jel a karater mezőben történő jobbra igazítására használható. A mező szélességét a # és = karakterek számával jelölhetjük ki a formátum mezőben. Ha a karakterlánc kevesebb karaktert tartalmaz mint a mező szélessége, a karaktersorozat jobbról tölti föl a mezőt. Ha viszont több karaktert tartalmaz, mint amennyi a mezőbe fér, a jobb szélső karakterek csonkulnak, és a karakterlánc az egész mezőt kitölti.

PUDEF

PUDEF "1 − 4 karakter"

A PUDEF lehetővé teszi, hogy legfeljebb 4 szimbólumot újra definiájunk a PRINT USING utasításban. Megváltoztathatjuk a szóközöket, vesszőket, tizedespontokat és dollár jeleket valamilyen más karakterre, ha az új karaktert megfelelő helyre írjuk a PUDEF kontrol mezejében.
Az első pozicló a kitöltő karakter. Alapértelmezés szerint szóköz. Írjunk ide új karaktert, ha a szóközök helyett mást akarunk megjeleníteni.
A második pozicló a vesszőt helyettesítő karakter helye.
Alapértelmezés szerint vessző.
A harmadik pozicló a tizedespont.
A negyedik a dollár.

```
10 PUDEF "*"        Szóköz
20 PUDEF " &"       A szóközök helyére csillagokat ír.
                    & jelet ír a vesszők helyett.
```



Pontot ír a vesszők, és tizedesvesszőt a tizedespontok helyett.
£ jelet ír a dollár helyett, pontot a tizedesvessző helyett, és tizedesvesszőt a tizedespont helyett.
Szóköz

READ

READ változó lista

Ezzel az utasítással a DATA utasításokban szereplő elemek közül adhatunk értéket a READ utasításban lévő változóknak. A READ utasítás változói között numerikus és karakterváltozók is szerepelhetnek. Ügyelni kell arra, hogy ne álljon ott karakter, ahol a READ utasítás számot vár, mert hibaüzenetet kapunk.
Példa:
READ A$,G$,Y

REM

REM üzenet

A REM utasítás a programlista olvasójának szánt megjegyzést tartalmaz. Lehet a program egy részéhez fűzött magyarázat, felvilágosítást nyújthat a szerzőről, stb. A REM utasítások nem befolyásolják a program működését, csupán hosszabbá teszik (és ennél fogva lelassítják azt). A REM utasítás után bármilyen szöveg következhet, bár a grafikus karakterek használata furcsa eredményt adhat.
Példa:
10 NEXT X: REM EZ A SOR FÖLÖSLEGES

RESTORE

RESTORE [sorszám]

Az utasítás végrehajtása után az adatpointer a soron következő DATA utasításban lévő soron következő adat helyett az első helyen álló adatra fog mutatni. Így lehetővé válik az adatok újraolvasása. Ha a RESTORE utasítást

sorszám követi, a pointer az adott sorra vonatkozik. Egyébként a pointer a programban lévő első DATA utasításra áll vissza.
Példa:
RESTORE 200

RESUME

RESUME [sorszám | NEXT]

Segítségével visszatérhetünk a programba egy hiba esetén kiadott TRAP utasítás után. Argumentum nélkül a RESUME megkísérli újra végrehajtani a sort, amelyben a hiba föllépett. A RESUME NEXT a program végrehajtását a hibás utasítást követő utasításnál folytatja, míg a RESUME sorszám utasítás esetén a kijelölt soron folytatódik a program végrehajtása.

RETURN

Ezt az utasítást mindig a GOSUB utasítással együtt használjuk. Ha a programban RETURN utasítás fordul elő, az utoljára végrehajtott GOSUB utasítást követő utasításra adódik a vezérlés. Ha előzőleg nem szerepelt GOSUB, a program végrehajtása megszakad, és a következő hibaüzenetet kapjuk: RETURN WITHOUT GOSUB ERROR (RETURN GOSUB nélkül).

SCALE

SCALE <1/0>

Többszínű és nagy felbontóképességű grafikus üzemmódban a bit térkép beosztását meg lehet változtatni a SCALE utasítással.
SCALE 1
Bekapcsolja a módosítást. Ezután a koordinátákat 0-tól 1023-ig kell megadni a szokásos x és y értékek helyett. A szokásos értékek a következők:

Többszínű üzemód ..... X = 0 – 159, Y = 0 – 199
Nagy felbontóképességű üzemmód ..... X = 0–319, Y = 0–199

SCALE 0 visszaállítja a szokásos koordinátákat.

SCNCLR

Törli a képernyőt mind grafikus, mind szöveg üzemmódban, illetve mindkettőben (osztott képernyő esetén).

SOUND

SOUND szólam, hangmagasság, időtartam

Ez az utasítás hangot állít elő a három szólam egyikét használva, a 0 — 1023 tartományban lévő frekvencia vezérléssel, és 0-tól 65535-ig terjedő időtartam meghatározással (hatvanad másodpercben).

SZÓLAM	
1	1.szólam (zenei hang)
2	2.szólam (zenei hang)
3	2.szólam (fehér zaj)

Ha egy újabb hangot kívánunk megszólaltatni, s ugyanazon szólam hangja még szól, a BASIC megvárja, míg az előző hang befejeződik. A nulla időtartammal kiadott SOUND utasítás speciális eset. Ilyenkor a BASIC azonnal kikapcsolja az éppen szóló hangot tekintet nélkül a még hátra lévő időre. Lásd a függelékben lévő zenei táblázatot, mely a zenei hangoknak megfelelő frekvencia értékeket tartalmazza.
Példa:
SOUND 2, 800, 3600 — **A második szólam használatával egy hangot szólaltat meg, melynek frekvencia értéke 800 és időtartama 1 perc.**

SSHAPE/GSHAPE

Az SSHAPE és GSHAPE utasításokkal BASIC karaktervváltozók segítségével többszínű vagy nagy felbontóképességű képernyők derékszögű területeit tárolhatjuk, illetve jeleníthetjük meg. Egy adott terület tárolására vonatkozó utasítás a következő:

SSHAPE karaktervváltozó, a1,b1 [,a2,b2]
Karaktervváltozó .....A karakterlánc neve, amiben adatot kívánunk megőrizni

a1,b1 .....Sarok koordináta
a2,b2 .....Az a1,b1-el szemben lévő csúcs koordinátája (alapértelmezés szerinti értéke a PC)

Mivel BASIC-ben a karakterláncok maximális hossza 255 karakter, a megőrízhető terület nagysága korlátozott. A szükséges karakterlánc méretét a következő képletek valamelyikével számolhatjuk ki:

L(többsz.) = INT((ABS(a1-a2) + 1) / 4 + .99)*(ABS(b1-b2) + 1) + 4
L(nagyf.) = INT((ABS(a1-a2) + 1) / 8 + .99)*(ABS(b1-b2) + 1) + 4

A többsz. többszínű, a nagyf. nagy felbontóképességű grafikus üzemmódot jelent.
A gép az alakzatot sorról sorra tárolja. A karakterlánc utolsó négy byte-ja az oszlop és sor hosszánál eggyel kisebb értéket tárol (ABS (a1-a2)-t) alacsony/magas helyértékű byte formátumban.
Egy tárolt alakzatnak a képernyő tetszés szerinti részén történő megjelenítésére szolgáló utasítás a következő:

GSHAPE karakterváltozó [,[a,b][,üzemmód]

Karakterváltozó .....A megjelenítésre szánt alakzatot tartalmazza.
a,b .....A bal felső koordináta, mely az alakzat helyét határozza meg (arányos – alapértelmezés szerinti értéke a PC)
Üzemmód .....Megjelenítés módja:
0: az alakzatot változtatás nélkül helyezi el (alapértelmezés)
1: inverz megjelenítést eredményez
2: az alakzat és a terület OR kapcsolata
3: az alakzat és a terület AND kapcsolata
4: az alakzat és a terület XOR kapcsolata
Példák:

SSHAPE "ADAT $",0,0 — A képernyőrészt a bal felső saroktól a kurzor pozíciójáig ADAT $ névén tárolja.

GSHAPE "ADAT $",,,1 — Megjeleníti az ADAT $ nevű alakzatot fordított háttér és előtér színnel, az alakzat bal felső csúcsával a kurzornál.

STOP

Ez az utasítás megállítja a programot. Egy BREAK IN LINE#; üzenet jelenik meg a képernyőn a program megállításakor, ahol a sorszám a STOP utasítást tartalmazó sor száma. A programot a STOP utasítást követő utasítástól a CONT paranccsal újraindíthatjuk. A STOP utasítást általában hibakereséskor használjuk.

SYS

SYS cím

A SYS utasítást decimális szám, vagy numerikus változó követi, melynek értéke 0-tól 65535-ig terjed. A program a gépi nyelvű program végrehajtását a megadott címtől kezdi. Az USR függvényhez hasonlóan működik, de paramétert nem ad át. Lásd Commodore 16 Programozói Hivatkozási Kézikönyvben a gépi nyelvű programokról írottakat!

TRAP

TRAP [sorszám]

Bekapcsoláskor a TRAP utasítás átveszi a hibák kezelését (beleértve a RUN/STOP billentyűt), a nem létező utasítás hiba, "UNDEFINED STATEMENT ERROR" kivételével. Végrehajtási hiba esetén a hibajelzés beállítása megtörténik, majd a vezérlés a TRAP utasításban megadott sorszámra kerül. A hibás sor sorszáma az EL rendszerváltozóban található. Az adott hibára vonatkozó hibakódot az ER nevű rendszerváltozó tartalmazza. Az ERR$ (ER) karakterváltozó az ER hibakódnak megfelelő hibaüzenetet tartalmazza.
MEGJEGYZÉS: a TRAP rutinban keletkezett hibára a TRAP utasítás hatástalan. A RESUME utasítás használható a végrehajtás folytatására. Ha a TRAP sorszám nélkül áll, kikapcsolja a TRAP utasítást.

TRON

A TRON utasítást hibakereséskor használjuk. Az utasítás elindítja a nyomkövetést. Nyomkövető üzemmódban minden végrehajtott utasítás sorszáma megjelenik a képernyőn.

TROFF

Ez az utasítás kikapcsolja a nyomkövető üzemmódot.

VOL

VOL hangerő

Beállítja a hangerőt a SOUND utasításban. A hangerőt 0 és 8 között állíthatjuk, ahol nyolc a maximális hangerő, és 0 a kikapcsolás. A VOL mindkét szólamra vonatkozik.

WAIT

WAIT cím, 1. érték [,2. érték]

A WAIT utasítással a program végrehajtása felfüggeszthető, míg egy memóriacím tartalma bizonyos módon megváltozik. A cím 0-tól 65535-ig terjedhet. Az értékeknek 0 és 255 közé kell esniük.
Az adott memóriacím tartalma először KIZÁRÓ VAGY kapcsolatba kerül a második értékkel (ha van ilyen), majd logikai ÉS kapcsolatba az első értékkel. Ha az eredmény nem nulla, a program a következő utasítással folytatódik.

További ismeretek a grafikus utasításokról

Van néhány fogalom mely minden bit-térképes grafikai utasításra vonatkozik. Először is a Pixel Kurzor (PC) fogalma. A PC a szöveg üzemmód kurzorjához hasonló; a következő pont ott jelenik meg, ahol a PC található. A szöveg kurzortól eltérően a PC láthatatlan. Minden rajzoló utasítás használja a PC-t, a LOCATE (elhelyezés) utasítás pedig lehetővé teszi a PC áthelyezését rajzolás nélkül is. A rajzoló utasításokban X,Y koordináták helyett bárhol használhatunk RELATIV koordinátákat is. A relatív koordináták a PC pillanatnyi poziciójától számítanak.
A relatív koordináták használatához helyezzünk plusz vagy mínusz jelet a koordináták elé. Az X érték előtt álló plusz jel a PC-t jobbra mozgatja, a

mínusz jel pedig balra. Hasonlóan az Y koordináta előtt álló mínusz jel felfelé, míg a plusz jel lefelé mozgatja a PC-t.

Például:

LOCATE + 100,-25 — 100-al jobbra, és 25-el felfelé mozgatja a PC-t.

DRAW 1,+ 10,+ 10TO100,100 — A PC pillanatnyi helyzeténél 10-zel jobbra és 10-zel lejjebb lévő pontból indítva egyenest rajzol a 100,100 abszolut koordinátájú pontba.

Kijelölhetünk távolságot és relatív szöget is a PC pillanatnyi helyzetéhez viszonyítva, a két paramétert pontosvesszővel választva el egymástól.

Például:

LOCATE 50;45 — A PC-t pillanatnyi helyzetéből 50 pontnyi távolságra 45 fokkal elmozdítja.

FÜGGVÉNYEK

Numerikus függvények

A numerikus függvényeket azért nevezzük így, mert eredményül számértéket adnak. A függvénykapcsolat a matematikai függvények kiszámításától képernyő pozició kijelöléséig terjed. A numerikus függvények formája a következő:

FÜGGVÉNY (paraméter)

A paraméter lehet szám, változó, kifejezés vagy karakter.

ABS(X) (abszolút érték)

Az abszolut érték függvény a paraméter pozitív értékét adja eredményül.

ASC(X$)

Ez a függvény az X$-ban tárolt karakterlánc első karakterének ASCII kódját adja vissza.

ATN(X) (arkusz tangens)

Megadja azt a szöget radiánban mérve, melynek tangense

COS(X) (koszinusz)

A radiánban mért X szög koszinuszát adja.

DEC ("hexadecimális karakterek")

Négy jegyű hexadecimális számokat decimálisra konvertál. (0 <hexadecimális számok < FFFF)
Példa:
N= DEC("F4")

EXP(X)

e (2.71828183) x-edik hatványát adja.

FNxx (X)

Megadja a DEF FNxx utasításban definiált xx függvény értékét.

INSTR (1. karakterlánc, 2. karakterlánc [,kezdő pozició])

A második karakterlánc helyzetét adja meg az első karakterláncon belül a kezdő poziciótól kezdve. A kezdő pozició alapértelmezés szerint az első karakterlánc kezdete. Ha nincs egyezés, 0-át ad eredményül. Példa:
PRINT INSTR "A MACSKA A KALAPBAN VAN", "MACSKA"
Az eredmény 3, mivel a macska első karakterlánc harmadik karakterén kezdődik.

(INT(X) (egész rész)

Az X egész részét adja. Az eredmény mindig kisebb X-nél, vagy egyenlő vele. Így a negatív tizedes számok egész része nem azonos az eredeti szám tizedesponttól balra lévő értékével. (pl. INT (– 4.5) = – 5.)

Ha az INT függvényt fel- vagy lekerekítésre használjuk, akkor a formátum a következő: INT(x+ .5) vagy INT (x– .5)
Példa:

INT(X*100 + .5)/100	A következő legmagasabb fillérre kerekít.

JOY(n) (botkormány)

Ahol n = 1 az 1-es botkormány poziciót, n = 2 a 2-es botkormány poziciót jelenti.
128 vagy annál nagyobb érték a tűz gomb lenyomását jelzi. Az irány jelzése a következőképpen történik:

FENN

tüzelés = 128 + 1

8 2

BAL 7 0 3 JOBB

6 4

5

LENN

Példa:
Ha JOY(2) 135-ös értéket kap, akkor a 2-es számú botkormány balra tüzel.

LOG(X) (logaritmus)

A függvény az X változó természetes logaritmuát adja. A természetes logaritmus az e alapú logaritmus. (Lásd az EXP(X)-et). Tizes alapú logaritmusra való átszámoláshoz (LOG(10)-zel kell osztani.

PEEK(x)

Ez a függvény az X memória cím tartalmát adja, ahol X 0 és 65535 közé esik, az eredmény pedig 0 és 255 között van. Gyakran használjuk a POKE utasítással együtt.

RCLR(N)

Az N-nek adott szint adja vissza (0=<N=<4)
(0 = háttér, 1 = előtér, 2 = többszínű 1., 3 = többszínű 2., 4 = keret)

RDOT(N)

A PC (Pixel Kurzor) pillanatnyi pozícióját adja.

N:0 XPOS értéke
N:1 YPOS értéke
N:2 szín forrás

RGR(X)

A pillanatnyi grafikus üzemmód kódját adja (X látszólagos paraméter, bármilyen érték lehet).

RLUM(N)

Az N forráshoz rendelt pillanatnyi fényerő szintet adja meg.

RND(X) (véletlen szám)

Ez a függvény véletlen számot ad 0 és 1 között. Alkalmas játékok esetén kockadobás és egyéb véletlenszerű események szimulálására, továbbá néhány statisztikai alkalmazásban is használható. Az első véletlen számot az RND (-TI) függvénnyel ajánlatos létrehozni, hogy a generálás mindig más számmal kezdődjön. Ezután az X 1, vagy bármely pozitív szám lehet. (X az a mag amire a véletlen számok épülnek.) Ha X nulla, a véletlen számok generálása RND minden végrehajtásakor az órából vett értékről indul. Ha X negatív, a véletlen számok generálása ezen a számon alapszik. Ugyanazon negatív szám használata ugyanazt a véletlen számsorozatot eredményezi. Ha X pozitív, a véletlen számok generálása a korábbi X értéktől függ.
Egy kockadobást a következő utasítással szimulálhatunk:

INT(RND(1)*6+ 1).

Először a 0 – 1 között lévő véletlen számot megszorozzuk 6-tal, ekkor a szám 0 és 6 közé esik (nagyobb mint 0, és kisebb mint 6). Ezután hozzáadunk egyet, hogy az érték 1-től 7-ig terjedjen. Az INT függvény levágja a tizedes jegyeket, így az eredmény 1 és 6 közé eső egész szám lesz. Két kockával való dobás szimulálásához adjunk össze két hasonlóan nyert számot.
Példa:

X= INT(RND (1)*6)+ INT(RND(1)*6)+ 2	Két kockát szimulál.
X= INT (RND(1)*1000)+ 1	1-től 1000-ig terjedő számot ad.
X= INT(RND(1)*150)+ 100	100 és 249 közé eső számot ad.

SGN(X) (előjel)

Ez a függvény X előjelét adja. Az eredmény + 1 ha X pozitív, 0 ha zéró, és – 1 ha negatív.

SIN(X) (színusz)

Trigonometrikus színusz függvény. Az eredmény a radiánban mért X szög színusza.

SQR(X) (négyzetgyök)

Ez a függvény X négyzetgyökét adja, ahol X pozitív szám, vagy nulla. Ha X negatív, a következő hibaüzenet jelenik meg: ILLEGAL QUANTITY ERROR (érvénytelen mennyiség).

TAN(X) (tangens)

A függvény a radiánban mért X szög tangensét adja.

USR(X)

Ha ezt a függvényt végrehajtjuk, a program egy gépi nyelvű programra ugrik, amelynek kezdő címét az 1281-es és 1282-es memória címek tartalmazzák. Az X paramétert a gépi nyelvű program a lebegő pontos akkumulátorból veszi át. A hívó változón keresztül egy másik paramétert küld vissza a BASIC programnak. Ezáltal lehetővé válik paraméterek cseréje gépi nyelvű programok és a BASIC között. Lásd Commodore 16 Programozói Hivatkozási Kézikönyvet az USR függvényre és a gépi nyelvű programozásra vonatkozó részletekkel kapcsolatosan.

VAL(X$)

Ez a függvény az X$-ban tárolt karakterláncot számmá alakítja, és lényegében az STR$ inverz művelete. A karakterláncot balról jobbra vizsgálja, és az első érvénytelen karakterig minden felismerhető numerikus karaktert számmá alakít. Ha a Commodore 16 érvénytelen karaktert talál, a karakterláncnak csak az utolsó érvényes karakterig terjedő részét konvertálja.
Példa:

10 X= VAL("123.456")	X= 123.456
10 X= VAL("3E03")	X= 3000
10 X= VAL("12A13B")	X= 12
10 X= VAL("RIUO17*")	X= 0
10 X= VAL("-1.23.23.23")	X= -1.23

Karakterlánc függvények

A karakterlánc függvények abban különböznek a numerikus függvényektől, hogy szám helyett karaktereket, grafikus jeleket vagy (idézőjelek között lévő) karakterláncból vett számjegyeket adnak eredményül.

CHR$(X)

Ez a függvény azt a karaktert adja, melynek ASCII kódja X.

ERR$(N)

Az N hibát leíró hibaüzenetet tartalmazza (lásd TRAP).

HEX$(N)

4 jegyű karakterláncot ad, mely N értékének hexadecimális megfelelője (0 < N <65535).

LEFT$(X$,X)

Az X$ bal oldali X karakteréből álló karaktersorozatot adja eredményül.

LEN(X$)

Az X$ karakterlánc karaktereinek számát adja (beleértve a szóközöket és egyéb jeleket is).

MID$(X$,N,X)

Ez a függvény X karakterből álló karakterláncot ad eredményül, amely X$ N-edik karakterén kezdődik. A MID$ állhat az értékadó kifejezés bal oldalán mint pszeudó változó, de lehet függvény is. MID$ (karakterváltozó, kezdő pozició, hossz) = forrás karakterlánc.
Ez a függvény helyettesíti a forrás karakterlánc karaktereit a kezdő pozíciótól a kezdet + hossz által meghatározott pozicióig a karakterváltozó karaktereivel. A hossz alapértelmezés szerint a karakterváltozó hossza, és hiba lép föl, ha a hossz nagyobb a forrás karakterlánc hosszánál.
Példa:

```
10 A$= "KÉSŐI VISZONTLÁTÁS"
20 PRINT A$
30 MID$(A$,2,3)= "ORA"
40 PRINT A$
```

RIGHT$(X$,X)

Az X$ jobb oldali X karakteréből álló karaktersorozatot adja eredményül.

STR$(X)

Olyan karakterláncot ad eredményül, ami megegyezik X kinyomtatott tartalmával.
Példa:

```
A$= STR$(X)
```

EGYÉB FÜGGVENYEK

FRE (X)

Ez függvény a memóriában hozzáférhető, fel nem használt byte-ok számát adja meg. X látszólagos argumentum.

POS(X)

Ez a függvény annak az oszlopnak a számát adja meg (0-39), melyben a következő PRINT utasítás kezdődik a képernyőn. X látszólagos argumentum.

SPC(X)

Ez a függvény PRINT utasításban használható X szóköz átugrására. X értéke 0 és 255 között lehet.

TAB(X)

Ez a függvény PRINT utasításban használható. A következő kinyomtatandó karakter az X számú oszlopban fog megjelenni. X értéke 0 és 255 között lehet.

π(PI)

A kifejezésekben alkalmazott PI értéke 3.14159265.

VÁLTOZÓK ÉS MŰVELETI JELEK

Változók

A Commodore 16 háromféle BASIC változót használ. Ezek a következők: numerikus valós, numerikus egész és karakter (alfanumerikus) változók.

A valós numerikus változók (melyeket lebegőpontos változóknak is nevezünk) értéke 10^{-38}–10^{+38}-ig terjed, pontosságuk 9 számjegy. Ha a szám 9 jegynél hosszabbá válik, azaz 10^{-10}-t vagy 10^{+10}-t túllépi, a számítógép tudományos jelölésmódra tér át, a szám 1 egész részből és 8 tizedes helyből áll, majd az E betű és 10 megfelelő hatványa követi. Például, az 12345678901 szám a következő módon jelenik meg: 1.23456789E+ 10.
Egész változókat használhatunk, ha a szám + 32767-től -32768-ig terjed, és törtrészt nem tartalmaz. Az egész változók egész számokat tartalmaznak, például 5,10, vagy -100. Az egész számok tömbelemekként kevesebb helyet foglalnak el, mint a lebegőpontos számok.
A karakterváltozók karakter adatok, számok, betűk és egyéb más, a Commodore 16 által ismert karakterek tárolására használhatók. Karakterváltozó tartalma például: "Commodore 16".

AZONOSÍTÓK

Az azonosítók állhatnak egyetlen betűből, egy betűből és egy azt követő számból, vagy két betűből. Az azonosítók két karakternél hosszabbak is lehetnek, de csak az első két karakter számít.
Az egész típusú változókat az azonosító után írt % jellel azonosítjuk. A karakterváltozók azonosító jele a $.
Példák:

Numerikus azonosítók A, A5, BZ
Egész típusú azonosítók: A%, A5%, BZ%
Karakter azonosítók: A$, A5$, BZ$

A tömbök egy név alatt változók sorozatát tartalmazzák, és külön szám vagy számok azonosítják a tömb elemeit. A tömböket a DIM utasítással definiáljuk, és lehetnek lebegőpontos, egész vagy karakterváltozó tömbök. A tömbazonosító után zárójelben álló szám az adott változó száma – 1.

Példák: A(7), BZ%(11), A$(87)

A tömbök egynél több dimenziósak is lehetnek. Egy két dimenziós tömböt sorokból és oszlopokból álló tömbnek tekinthetünk, melyben az első szám az oszlopot, a második pedig a sort jelöli ki. (Mintha egy térkép hálózata lenne.)

Példák: A(7,2), BZ%(2,3,4), Z$(3,2)

RENDSZER-AZONOSÍTÓK

Hét olyan azonosító van, melyeket a Commodore 16 lefoglal, és más célra nem használhatók. Ezek a következők: DS, DS$, ER, EL, ST, TI, és TI$.
Nem használhatunk a BASIC kulcsszavakkal inegegyező azonosítókat sem, például TO, és IF, sőt olyan azonosítókat sem, melyek ilyen kulcsszavakat tartalmaznak, például SRUN, XLOAD, stb.
Az ST a kivitelre és bevitelre vonatkozó státusz változó (kivéve a normál képernyő/billentyűzet műveleteket). Az ST változó tartalma az előző bemeneti/kimeneti művelet eredményétől függ. Az ST változó részletes magyarázata a Commodore 16 Programozói Hivatkozási Kézikönyben található, de általában ST értéke 0, ha a művelet sikeres volt.
A TI és TI$ változók a Commodore 16-ba épített órára vonatkoznak. A rendszeróra minden 1/60-ad másodpercben frissül fel. A Commodore 16 bekapcsolásakor 0-ról indul, és csak TI$ értékének megváltozásával állítható be újra. A TI változó megadja az óra pillanatnyi értékét 1/60-ad másodpercben. A TI$ 24 órás beosztásban tartalmazza az időértékeket. A TI$ első két karaktere az órákat, a harmadik és negyedik a perceket, az ötödik és hatodik pedig a másodperceket tartalmazza. A változó bármilyen értékre beállítható (feltéve, hogy minden karaktere szám) és automatikusan 24 órás óraként működik.

Példa: TI$ = "101530" az órát 10:15-re és 30 másodpercre állítja.

Az óra értéke elvész, ha a Commodore 16-ot kikapcsoljuk. 0-ról indul, ha a számítógépet bekapcsoljuk, és újra nullázódik, ha az óra értéke a 235959-et meghaladja (23 óra, 59 perc és 59 másodperc).
A DS változó a lemezmeghajtó parancs-csatornáját olvassa be, és a meghajtó pillanatnyi státuszát adja vissza. Az üzenetet a PRINT DS$ utasítással jeleníthetjük meg. Ezeket az állapotváltozókat lemezműveletek, például a DLOAD vagy DSAVE után használhatjuk, ha a piros lámpa villog a lemezmeghajtón.
Az ER, EL és ERR$ változókat, hibaelemző rutinokban használhatjuk.
Általában csak programon belül használhatók. Az ER a program elindítása óta előfordult utolsó hiba számát tartalmazza. Az EL az a sor, amelyikben a hiba föllépett. Az ERR$ egy függvény, mely lehetővé teszi a BASIC hibaüzenetek kiiratását. A PRINT ERR$(ER) utasítás a megfelelő hibaüzenetet írja ki.

BASIC MŰVELETI JELEK

Az aritmetikai műveleteket a következő jelekkel hajthatjuk végre:

+ összeadás
- kivonás
* szorzás
/ osztás
↑ hatványozás

Ha egy sorban több kijelölt művelet van, a műveletek végrehajtására prioritásuk sorrendjében kerül sor. A műveletek közötti prioritás a következő: első a hatványozás, majd a szorzás és osztás következik, végül az összeadás és kivonás. Az azonos prioritású műveletek végrehajtására balról jobbra sorrendben kerül sor. Ha a műveleteket ettől eltérő sorrendben kívánjuk végrehajtani, a Commodore 16 BASIC-je lehetővé teszi, hogy egyes számításoknak zárójelezéssel nagyobb prioritást adjunk. A gép először a zárójelbe tett műveleteket fogja végrehajtani. Mindig ügyelni kell arra, hogy a kifejezésekben a nyitó és záró zárójelek száma megegyezzen, különben szintaktikai hibaüzenetet (SYNTAX ERROR) kapunk. Az egyenlőséget vagy egyenlőtlenséget kifejező műveleti jeleket relációs műveleti jeleknek nevezzük.

Az aritmetikai műveletek mindig magasabb prioritásúak a relációknál.

= Egyenlő
< Kisebb
> Nagyobb
<= ,= < Kisebb vagy egyenlő
>= ,= > Nagyobb vagy egyenlő
<>, >< Nem egyenlő

Végül létezik három olyan logikai művelet, melyek prioritása alacsonyabb az arimetikai és relációs műveletekénél is:

AND
OR
NOT

Ezeket leggyakrabban több kifejezés egyesítésére használjuk az IF...THEN utasításokban. Aritmetikai kifejezésekben végrehajtásukra az összeadás és kivonás után kerül sor.

Példák:

IF A= B AND C= D THEN 100	A= B és C= D állításnak is igaznak kell lennie.
IF A= B OR C= D THEN 100	A= B vagy C= D valamelyikének igaznak kell lennie.
A= 5:B= 4:PRINT A= B	0-át ír ki.
A= 5:B= 4:PRINT A >B	-1-et ír ki.
PRINT 123 AND 15:PRINT 5 OR 7	11-et és 7-et ír ki.

BASIC rövidítések és hivatkozási táblázat

KULCSSZÓ	RÖVIDÍTÉS	TÍPUS
ABS	a SHIFT B	numerikus függvény
ASC	a SHIFT S	numerikus függvény
ATN	a SHIFT T	numerikus függvény
AUTO	a SHIFT U	parancs
BACKUP	b SHIFT A	parancs
BOX	b SHIFT O	utasítás
CHAR	ch SHIFT A	utasítás
CHR$	c SHIFT H	karakter függvény
CIRCLE	c SHIFT I	utasítás
CLOSE	cl SHIFT O	utasítás
CLR	c SHIFT L	utasítás
CMD	c SHIFT M	utasítás
COLLECT	col SHIFT L	parancs
COLOR	co SHIFT L	utasítás
CONT	c SHIFT O	parancs
COPY	co SHIFT P	parancs
COS	nincs	numerikus függvény
DATA	d SHIFT A	utasítás
DEC	nincs	numerikus függvény
DEF FN	d SHIFT E	utasítás
DELETE	de SHIFT L	parancs
DIM	d SHIFT I	utasítás
DIRECTORY	di SHIFT R	parancs
DLOAD	d SHIFT L	parancs
DO	nincs	utasítás
DRAW	d SHIFT R	utasítás
DSAVE	d SHIFT S	parancs
END	e SHIFT N	utasítás
ERR$	e SHIFT R	karakter függvény
EXP	e SHIFT X	numerikus függvény
FOR	f SHIFT O	utasítás
FRE	f SHIFT R	numerikus függvény
GET	g SHIFT E	utasítás
GETKEY	getk SHIFT E	utasítás

KULCSSZÓ	RÖVIDÍTÉS	TÍPUS
GET #	nincs	utasítás
GOSUB	go SHIFT S	utasítás
GOTO	g SHIFT O	utasítás
GRAPHIC	g SHIFT R	utasítás
GSHAPE	g SHIFT S	utasítás
HEADER	he SHIFT A	parancs
HEX$	h SHIFT E	karakter függvény
IF...GOTO	nincs	utasítás
IF...THEN...ELSE	nincs	utasítás
INPUT	nincs	utasítás
INPUT#	i SHIFT N	utasítás
INSTR	in SHIFT S	numerikus függvény
INT	nincs	numerikus függvény
JOY	j SHIFT O	numerikus függvény
KEY	k SHIFT E	parancs
LEFT$	le SHIFT F	karakter függvény
LEN	nincs	numerikus függvény
LET	l SHIFT E	utasítás
LIST	l SHIFT I	parancs
LOAD	l SHIFT O	parancs
LOCATE	lo SHIFT C	utasítás
LOG	nincs	numerikus függvény
LOOP	lo SHIFT O	utasítás
MID$	m SHIFT I	karakter függvény
MONITOR	m SHIFT O	utasítás
NEW	nincs	parancs
NEXT	n SHIFT E	utasítás
ON...GOSUB	on...go SHIFT S	utasítás
ON...GOTO	on...g SHIFT O	utasítás
OPEN	o SHIFT P	utasítás
PAINT	p SHIFT A	utasítás
PEEK	p SHIFT E	numerikus függvény
POKE	p SHIFT O	utasítás
POS	nincs	numerikus függvény
PRINT	?	utasítás
PRINT#	p SHIFT R	utasítás
PRINT USING	?us SHIFT I	utasítás
PUDEF	p SHIFT U	utasítás
RCLR	r SHIFT C	numerikus függvény
RDOT	r SHIFT D	numerikus függvény
READ	r SHIFT E	utasítás
REM	nincs	utasítás

KULCSSZÓ	RÖVIDÍTÉS	TÍPUS
RENAME	re SHIFT N	parancs
RENUMBER	ren SHIFT U	parancs
RESTORE	re SHIFT S	utasítás
RESUME	res SHIFT U	utasítás
RETURN	re SHIFT T	utasítás
RGR	r SHIFT £G	numerikus függvény
RIGHT$	r SHIFT I	karakter függvény
RLUM	r SHIFT L	numerikus függvény
RND	r SHIFT N	numerikus függvény
RUN	r SHIFT U	parancs
SAVE	s SHIFT A	parancs
SCALE	sc SHIFT A	utasítás
SCNCLR	s SHIFT C	utasítás
SCRATCH	sc SHIFT R	parancs
SGN	s SHIFT G	numerikus függvény
SIN	s SHIFT I	numerikus függvény
SOUND	s SHIFT O	utasítás
SPC(	s SHIFT P	különleges függvény
SQR	s SHIFT Q	numerikus függvény
SSHAPE	s SHIFT S	utasítás
STATUS	nincs	fenntartott num. változó
STOP	s SHIFT T	utasítás
STR$	st SHIFT R	karakter függvény
SYS	s SHIFT Y	utasítás
TAB(	t SHIFT A	különleges függvény
TAN	nincs	numerikus függvény
TI	nincs	fenntartott num. változó
TI$	nincs	fenntartott karakterváltozó
TRAP	t SHIFT R	utasítás
TROFF	tro SHIFT F	utasítás
TRON	tr SHIFT O	utasítás
UNTIL	u SHIFT N	utasítás
USR	u SHIFT S	különleges függvény
VAL	nincs	numerikus függvény
VERIFY	v SHIFT E	parancs
VOL	v SHIFT O	utasítás
WAIT	w SHIFT A	utasítás
WHILE	w SHIFT H	utasítás

FÜGGELÉKEK

- Hibaüzenetek
- Lemez hibaüzenetek
- Matematikai függvények származtatása
- Zenei hangjegytáblázat
- Képernyő és ASCII kódok
- Irodalomjegyzék

A FÜGGELÉK

Hibaüzenetek

Ezeket a hibaüzeneteket a BASIC írja ki. A hibaüzenetek az ERR$() függvény alkalmazásával is kiírathatók. A hibakód a hibára vonatkozó szám, amit a függvény alkalmazásakor megadunk.

Hibakód	Hibaüzenet	
1	**TOO MANY FILES** (túl sok állomány van nyitva)	Egyidejűleg maximum 10 állomány lehet megnyitva.
2	**FILE OPEN** (az állomány meg van nyitva)	Kísérlet egy már megnyitott állomány újbóli megnyitására.
3	**FILE NOT OPEN** (az állomány még nincs nyitva)	A kimeneti/bemeneti utasításban szereplő állományt használat előtt meg kell nyitni.
4	**FILE NOT FOUND** (ilyen állomány nincs)	Az adott nevű állomány nem létezik (lemezen).
5	**DEVICE NOT PRESENT** (ilyen egység nincs)	A megadott kimeneti/bemeneti egység nem érhető el.
6	**NOT INPUT FILE** (nem bemeneti állomány)	GET-tel vagy INPUT-tal megkísérelt adatbevitel kimeneti állományból.
7	**NOT OUTPUT FILE** (nem kimeneti állomány)	Megkísérelt adatkivitel csak bemenetiként kezelt állományba.
8	**MISSING FILE NAME** (hiányzó állománynév)	Lemezre kiadott OPEN, LOAD vagy SAVE utasításokban általában szükség van az állománynévre.
9	**ILLEGAL DEVICE NUMBER** (érvénytelen egységcím)	Helytelen eszközhasználatra tett kísérlet. (SAVE a képernyőre, stb.)
10	**NEXT WITHOUT FOR** (NEXT nélküli FOR)	Hurkok hibás egymásba ágyazása, vagy a NEXT utasításban megadott változó nem egyezik a FOR-ban megadottal.
11	**SYNTAX** (szintaktikai hiba)	A BASIC nem ismeri fel az utsítást. Hiányzó vagy fölösleges zárójelek, elírt kulcsszavak, stb. miatt.
12	**RETURN WITHOUT GOSUB** (RETURN GOSUB nélkül)	Egy RETURN utasítás végrehajtására került sor előzetes GOSUB nélkül.

13	**OUT OF DATA** (túl kevés adat)	A READ utasítás nem talál több beolvasásra szánt adatot.
14	**ILLEGAL QUANTITY** (érvénytelen mennyiség)	Egy függvény vagy utasítás argumentumaként megadott szám kívül esik a megengedhető tartományon.
15	**OVERFLOW** (túlcsordulás)	Egy számítás eredménye nagyobb a legnagyobb ábrázolható számnál (1.701411833E+ 38).
16	**OUT OF MEMORY** (nincs szabad memória)	Nincs szabad tárterület a program vagy a változók tárolására, illetve túl sok DO, FOR vagy GOSUB utasítás van érvényben.
17	**UNDEF'D STATEMENT** (nemlétező utasítás)	Egy nem létező utasítássorra történt hivatkozás.
18	**BAD SUBSCRIPT** (helytelen tömbindex)	Olyan tömbelemre történt hivatkozás, mely a DIM utasításban megadott határon kívül esik.
19	**REDIM'D ARRAY** (tömb újra dimenzionálása)	Egy tömb csak egyszer dimenzionálható. Ha egy tömbre dimenzionálása előtt hivatkozunk, mérete automatikusan DIM (10)
20	**DIVISION BY ZERO** (nullával való osztás)	A nullával való osztás érvénytelen.
21	**ILLEGAL DIRECT** (érvénytelen parancs)	Az INPUT vagy GET utasítások csak programban használhatók.
22	**TYPE MISMATCH** (nem egyező típus)	Szám szerepel karakter helyett vagy fordítva.
23	**STRING TOO LONG** (túl hosszú karakterlánc)	Egy karakterlánc maximum 255 karaktert tartalmazhat.
24	**FILE DATA** (hibás adat)	Hibás adat olvasása szalagról.
25	**FORMULA TOO COMPLEX** (túl bonyolult képlet)	Egyszerűsíteni kell a kifejezést, (két részre bontani vagy kevesebb zárójelet alkalmazni).
26	**CAN'T CONTINUE** (nem folytatható)	A CONT parance érvénytelen, ha a program még nem futott, hiba fordult elő vagy egy sort módosítottak.
27	**UNDEF'D FUNCTION** (nem definiált függvény)	Felhasználó által definiált függvényre való hivatkozás a függvény definiálása előtt.
28	**VERIFY** (eltérés)	A lemezen vagy szalagon lévő program nem egyezik a memóriában lévővel.
29	**LOAD** (betöltési hiba)	Betöltés közben föllépett hiba.

30	**BREAK** (megszakítás)	A program végrehajtása a stop billentyűvel megszakítva.
31	**CAN'T RESUME** (TRAP nélküli RESUME)	TRAP utasítás kiadása nélkül előfordult RESUME.
32	**LOOP NOT FOUND** (LOOP nélküli DO)	A programban előfordult DO utasításhoz nem tartozik LOOP.
33	**LOOP WITHOUT DO** (DO nélküli LOOP)	Aktív DO utasítás nélkül előfordult LOOP.
34	**DIRECT MODE ONLY** (csak parancsmódban)	A parancs csak közvetlen módban használható, programban nem.
35	**NO GRAPHICS AREA** (nincs grafikus terület lefoglalva)	Grafikus utasítások (DRAW, BOX, stb.) előfordulása előzetes GRAPHIC utasítás nélkül.
36	**BAD DISK** (rossz lemez)	Lemezinicializálási hiba a gyors (ID nélküli) módszer előzőleg nem formázott lemezen való alkalmazása vagy hibás adathordozó miatt.

B FÜGGELÉK

Lemez hibaüzenetek

Ezeket a hibaüzeneteket a DS és DS$ fönntartott változók segítségével lehet megjeleníteni.
MEGJEGYZÉS: a 20-nál kisebb hibaüzenetek elhanyagolhatók a 01 kivételével, mely a SCRATCH utasítással törölt állományok számára vonatkozik.

20	**READ ERROR**	A lemezvezérlő nem tudta azonosítani a keresett adatblokk fejlécét. Az ok érvénytelen szektorszám vagy a fejléc megsérülése lehet.
21	**READ ERROR**	A lemezvezérlő nem talált szinkronjelet a keresett sávon. Oka lehet az író/olvasófej helytelen beállítása, az adathordozó hiánya, egy nem formázott vagy helytelenül behelyezett lemez. Jelezhet géphibát is.
22	**READ ERROR**	A lemezvezérlő egy nem megfelelően felírt adatblokkot olvasott vagy ellenőrzött. Ez a hibaüzenet a BLOCK

		parancsok használatakor fordul elő, és érvénytelen sáv/szektor olvasását jelzi.
23	**READ ERROR**	Ez az üzenet azt jelzi, hogy egy vagy több adatbyte hibás. Az adatok beolvasása a DOS memóriájába megtörtént, de az adatok után beolvasott ellenőrző összeg hibás. Utalhat földelési hibára is.
24	**READ ERROR**	Az adat vagy fejléc beolvasása a DOS memóriájába megtörtént, de egy hardver hiba az adat-mezőben érvénytelen bitkombináció beolvasását jelzi.
25	**WRITE ERROR**	Ez az üzenet akkor fordul elő, ha a vezérlő ëltérést talál a felírt és a DOS memóriájában lévő adatok között.
26	**WRITE PROTECTION ON**	Ez az üzenet akkor fordul elő, ha a vezérlő írás utasítást kapott, miközben a lemezen az írás-engedélyező rés le van ragasztva.
27	**READ ERROR**	A vezérlő a keresett adatblokk fejlécében hibát talált. Az adat beolvasására nem kerül sor. Utalhat földelési hibára is.
28	**WRITE ERROR**	A vezérlő egy adatblokk felírása után a következő fejléc szinkronjelére vár. Ha a szinkronjel nem jelenik meg az előre megadott időn belül, ez a hibaüzenet fordul elő. A hiba rossz lemezformázás (az adat belenyúlik a következő blokkba) vagy hardver hiba miatt lép föl.
29	**DISK ID MISMATCH**	Ez az üzenet akkor fordul elő, ha a vezérlő egy nem inicializált lemezt akár elérni. A hiba akkoris fölléphet, ha a lemez fejléce hibás.
30	**SYNTAX ERROR**	A DOS a parancscsatornán küldött parancsot nem tudja értelmezni. Rendszerint érvénytelen számú vagy formájú állománynév megadásakor fordul elő. Például két állománynév szerepel a COPY parancs bal oldalán.

31	**SYNTAX ERROR**	A DOS nem ismeri föl a parancsot. A parancsnak az első oszlopbèn kell kezdődnie.
32	**SYNTAX ERROR**	A kiküldött parancs 58 karakternél hosszabb.
33	**SYNTAX ERROR**	Az OPEN vagy SAVE parancsban érvénytelen paraméterek szerepelnek.
34	**SYNTAX ERROR**	A parancsból hiányzik az állománynév, vagy a DOS nem ismeri föl. Rendszerint a kettőspont (:) hiányzik a parancsból.
39	**SYNTAX ERROR**	Akkor fordulhat elő, ha a parancscsatornán (a másodlagos cím 15) küldött parancsot a DOS nem ismeri föl.
50	**RECORD NOT PRESENT**	Az utolsó rekordon túl INPUT# vagy GET# parancsokkal megkísérelt olvasás eredménye. Az üzenet a relatív állomány vége utáni pozicionáláskor is előfordul. Ha a cél az állomány kiterjesztése új rekordok hozzáadásával (a PRINT# paranccsal), a hibaüzenet figyelmen kívül hagyható. A hiba fellépésekor INPUT vagy GET parancsot csak újrapozicionálás után lehet kiadni.
51	**OVERFLOW IN RECORD**	A PRINT# parancs végrehajtáskor túllépi a rekord határát. A ki nem írt adat elvész. Mivel a rekord végjeleként küldött kocsi vissza is beleszámít a rekord méretébe, a hiba akkor fordul elő, ha a rekord összes karaktereinek száma (beleértve a záró kocsi visszát), túllépi a megadott méretet.
52	**FILE TOO LARGE**	Relatív állományon belül a rekord pozíciója a lemez túlcsordulását eredményezi.
60	**WRITE FILE OPEN**	Az üzenet egy írás után le nem zárt állomány olvasás előtti megnyitásakor fordul elő.
61	**FILE NOT OPEN**	Az üzenet egy meg nem nyitott állomány megkísérelt elérésekor fordul elő. Az üzenet nem mindig jele-

		nik meg, csak a végrehajtásra egyszerűen nem kerül sor.
62	**FILE NOT FOUND**	A keresett állomány nem található a jelzett meghajtón.
63	**FILE EXISTS**	A létrehozni kívánt állomány már szerepel a lemezen.
64	**FILE TYPE MISMATCH**	A keresett állomány típusa nem egyezik a tartalomjegyzékben szereplő állomány típusával.
65	**NO BLOCK**	Ez az üzenet a B-A parancs használatakor fordul elő. Azt jelzi, hogy a lefoglalandó blokk már foglalt. A paraméterek a következő szabad sáv és szektor címét tartalmazzák. Ha a paraméterek értéke 0, minden magasabb számú blokk foglalt.
66	**ILLEGAL TRACK AND SECTOR**	A DOS a formátumban nem létező sáv vagy blokk elérését kísérelte meg. A következő blokk mutatójának olvasási hibája eredményezheti.
67	**ILLEGAL SYSTEM T OR S**	Ez a különleges hibaüzenet a rendszer által használt érvénytelen sáv- vagy szektor-címet jelez.
70	**NO CHANNEL**	A keresett csatorna nem érhető el, vagy minden csatorna használatban van. Egyidejűleg maximum 5 soros állomány lehet megnyitva a DOS-ban. A közvetlen elérésű csatornák hat nyitott állomány elérését teszik lehetővé.
71	**DIRECTORY ERROR**	A BAM nem egyezik a belső számlálóval. A hiba a BAM foglaltságában van, vagy törlődött a DOS memóriájában. A hiba kiküszöböléséhez helyre kell állítani a BAM-ot a memóriában, az I parancs végrehajtásával. A hiba kiküszöbölése lezárhatja a nyitott állományokat. MEGJEGYZÉS: BAM = Block Availability Map (elérhető blokktérkép)
72	**DISK FULL**	Elfogytak a lemezen lévő blokkok vagy a tartalomjegyzék megtelt. A DISK FULL üzenet akkor jelenik meg az1541-en, amikor még két szabad

blokk van, ami lehetővé teszi az adott állomány lezárását.

73 DOS MISMATCH
(73, CBM DOS V2.6 1541)

A DOS 1-es és 2-es változatai csak olvasni tudják egymás lemezeit. A lemezeket kölcsönösen bármelyik változat tudja olvasni, de az egyik változat által formázott lemezre a másik nem tud írni, mivel a formátum különböző. Az üzenet a nem egyező formátum esetén tett íráskísérletnél jelenik meg. (Egy programrutin lehetővé teszi a formátum átkonvertálását.) Az üzenet bekapcsolás után is megjelenhet.

74 DRIVE NOT READY

Kísérlet egy lemezmeghajtó elérésére, melyben nincs lemez.

C FÜGGELÉK

Matematikai függvények származtatása

A BASIC 3.5-ben nem értelmezett függvények a következőképpen számíthatók ki:

FÜGGVÉNY	BASIC MEGFELELŐJE
Szekáns	SEC(X)= 1/COS(X)
Koszekáns	CSC(X)= 1/SIN(X)
Kotangens	COT(X)= 1/TAN(X)
Arkusz szinusz	ARCSIN(X)= ATN(X/SQR(- X*X+ 1))
Arkusz koszinusz	ARCCOS (X)= -ATN(X/SQR -X*X+ 1))+ π/2
Arkusz szekáns	ARCSEC(X)= ATN(X/SQR(X*X-1))
Arkusz koszekáns	ARCCSC(X)= ATN(X/SQR(X*X-1)) + (SGN(X)-1*π/2)
Arkusz kotangens	ARCOT(X)= ATN(X)+ π/2
Szinusz hiperbolikusz	SINH(X)= (EXP(X)-EXP(-X))/2
Koszinusz hiperbolikusz	COSH(X)= (EXP(X)+ EXP(-X))/2
Tangens hiperbolikusz	TANH(X)= EXP(-X)/(EXP(x)+ EXP (-X))*2+ 1

```
Szekáns hiperbolikusz                  SECH(X)= 2/(EXP(X)+ EXP(-X))
Koszekáns hiperbolikusz                CSCH(X)= 2/(EXP(X)-EXP(-X))
Kotangens hiperbolikusz                COTH(X)= EXP(-X)/(EXP(X)
                                         -EXP(-X))*2+ 1
Arkuszszinusz hiperbolikusz            ARCSINH(X)= LOG(X+ SQR(X*X+ 1))
Arkuszkoszinusz hiperbolikusz          ARCCOSH(X)= LOG(X+ SQR(X*X-1))
Arkusztangens hiperbolikusz            ARCTANH(X)= LOG(1+ X)/(1-X))/2
Arkusszekáns hiperbolikusz             ARCSECH(X)= LOG((SQR
                                         (-X*X+ 1)+ 1)/X)
Arkuszkoszekáns hiperbolikusz          ARCCSCH(X)= LOG((SGN(X)*SQR
                                         (x*x+ 1)/x))
Arkuszkotangens hiperbolikusz          ARCCOTH(X)= LOG((X+ 1)/(X-1))/2
```

D FÜGGELÉK

Zenei hangjegytáblázat

HANGJEGY	HANGREGISZTER ÉRTÉKE	TÉNYLEGES FREKVENCIA (Hz)
A	16	111.0
H	127	124.7
C	177	132.0
D	269	148.1
E	351	166.2
F	389	176.1
G	458	197.6
A	521	222.3
H	575	249.1
C	600	263.8
D	646	295.9
E	688	332.9
F	707	352.8
G	741	395.2
A	772	443.8
H	800	499.3

HANGJEGY	HANGREGISZTER ÉRTÉKE	TÉNYLEGES FREKVENCIA (Hz)
C	812	527.5
D	836	594.9
E	856	665.7
F	865	703.4
G	882	787.6
A	898	887.6
H	912	998.6
C	918	1055.0
D	930	1189.8
E	940	1331.4
F	945	1415.7
G	954	1597.7

A fenti táblázat négy oktávnyi hangjegy hangregiszter értékeit tartalmazza. A hangregiszter értékei adják a SOUND utasítás második paraméterét. A táblázatban található első hang megszólaltatásához (A — hangregiszter 7) a 7-et a SOUND utasítás második paramétereként kell megadni: SOUND 1,7,30.
A táblázatban nem található hangok megszólaltatásához szükséges értékeket a következő képlettel számolhatjuk:

Hangregiszter értéke = 1024-(111840.45/frekvencia)

E FÜGGELÉK

Képernyőkódok

A következő táblázat a Commodore karakterkészletekben található karaktereket tartalmazza. Ezeket az értékeket kell a képernyőmemóriába (3072-4095) POKE-olni a kívánt karakter megjelenítéséhez. (Ne feledkezzünk meg a színmemória beállításáról sem! — 2048-3071) A táblázat az egyes karakterekhez tartozó PEEK értékeket is tartalmazza.

Két karakterkészlet áll rendelkezésre, de egyidejűleg nem érhetők el. Ez azt jelenti, hogy miközben az egyik készlet karakterei láthatók a képernyőn, nem jeleníthetünk meg a másik készletből származó karaktereket. A karakterkészletek átkapcsolása a SHIFT és C= billentyűk egyidejű lenyomásával lehetséges.
BASIC-ből a PRINT CHR$(142) utasítással válthatunk a nagybetűs/grafikus üzemmódra, míg a PRINT CHR$(14) utasítással a nagybetűs/kisbetűs üzemmódra.
A táblázatban található bármely értéket inverz módban is megjeleníthetünk. Az inverz kódokat a táblázatból vett értékeknek és 128-nak az összeadásával nyerhetjük.

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
@		0	T	t	20	(		40
A	a	1	U	u	21	)		41
B	b	2	V	v	22	*		42
C	c	3	W	w	23	+		43
D	d	4	X	x	24	,		44
E	e	5	Y	y	25	-		45
F	f	6	Z	z	26	.		46
G	g	7	[		27	/		47
H	h	8	£		28	0		48
I	i	9	]		29	1		49
J	j	10	↑		30	2		50
K	k	11	←		31	3		51
L	l	12	SPACE		32	4		52
M	m	13	!		33	5		53
N	n	14	"		34	6		54
O	o	15	#		35	7		55
P	p	16	$		36	8		56
Q	q	17	%		37	9		57
R	r	18	&		38	:		58
S	s	19	'		39	;		59

SET 1	SET 2	POKE
<		60
=		61
>		62
?		63
		64
	A	65
	B	66
	C	67
	D	68
	E	69
	F	70
	G	71
	H	72
	I	73
	J	74
	K	75
	L	76
	M	77
	N	78
	O	79
	P	80
	Q	81
	R	82
	S	83
	T	84
	U	85
	V	86
	W	87
	X	88
	Y	89
	Z	90
		91
		92
		93
		94
		95
SPACE		96
		97
		98
		99
		100
		101
		102
		103
		104
		105
		106
		107
		108
		109
		110
		111
		112
		113
		114
		115
		116
		117
		118
		119
		120
		121
		122
		123
		124
		125
		126
		127

A 128-255-ig terjedő kódok a 0-127-ig terjedő kódok inverz megfelelői.

F FÜGGELÉK

ASCII és CHR$ kódok

Ez a táblázat azt mutatja, milyen karakterek jelennek meg a PRINT CHR$(X) utasítás hatására, X minden értékére. Feltünteti a PRINT ASC("X") utasítás hatására kapott értékeket is, ahol X bármely, a billentyűzeten található karakter lehet. A táblázat hasznos a GET utasítással kapott karakterek kiértékeléséhez, a nagybetű/kisbetű átalakításhoz és a karakterkódokon alapuló parancsok végrehajtásához, (például a nagybetű/kisbetű átkapcsoláshoz), ha a karaktert nem lehet idézőjelek között megadni.

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
	0	↑	17	"	34	3	51
	1	RVS ON	18	#	35	4	52
	2	HOME	19	$	36	5	53
	3	DEL	20	%	37	6	54
	4		21	&	38	7	55
WHT	5		22	'	39	8	56
	6		23	(	40	9	57
	7		24	)	41	:	58
TILTJA SHIFT C	8		25	*	42	;	59
ENGEDÉLYEZI SHIFT C	9		26	+	43	<	60
	10	ESCAPE	27	,	44	=	61
	11	RED	28	–	45	>	62
	12	→	29	.	46	?	63
RETURN	13	GRN	30	/	47	@	64
ÁTKAPCS. KISBETŰRE	14	BLU	31	0	48	A	65
	15	SPACE	32	1	49	B	66
	16	!	33	2	50	C	67

PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$	PRINTS	CHR$
D	68	♠	97	π	126	▦	155
E	69	▯	98	◥	127	PUR	156
F	70	▭	99		128	←	157
G	71	▭	100	♠	129	YEL	158
H	72	▭	101	FLASH ON	130	CYN	159
I	73	▭	102	FLASH OFF	131	SHIFT SPACE	160
J	74	▯	103		132	◧	161
K	75	▯	104	f1	133	▬	162
L	76	◹	105	f3	134	▢	163
M	77	◹	106	f5	135	▢	164
N	78	◹	107	f7	136	▢	165
O	79	▢	108	f2	137	▩	166
P	80	◹	109	f4	138	▢	167
Q	81	◹	110	f6	139	▬	168
R	82	▢	111	HELP	140	◪	169
S	83	▢	112	SHIFT RETURN	141	▯	170
T	84	●	113	ÁTKAPCS. NAGYBETŰRE	142	▦	171
U	85	▢	114		143	▙	172
V	86	♥	115	DEL	144	▙	173
W	87	▢	116	↓	145	▙	174
X	88	◹	117	INS OFF	146	▬	175
Y	89	☒	118	CLR	147	▙	176
Z	90	○	119	INST	148	▦	177
[	91	♣	120	◹	149	▦	178
£	92	▯	121	☒	150	▦	179
]	93	♦	122	○	151	◧	180
↑	94	▦	123	♣	152	◧	181
←	95	▩	124	▯	153	◨	182
▭	96	▯	125	♦	154	▭	183

A következő jegyzék válogatást közöl a kapható számítástechnikával és programozással foglalkozó könyvekről.

MAGYAR NYELVŰ IRODALOM

A BASIC programozási nyelv (Lőcs Gyula, Sarkadi Nagy István, Szlankó János) Műszaki Kvk., Bp. 1976.

Programozás BASIC nyelven (dr. Kocsis András) SZÁMALK, Bp. 1983.

A BASIC programozás technikája (Bodor– Gerő) SZÁMALK, Bp.

COMMODORE 64 I– II. (dr. Úry László) OMIKK LSI– ATSZ, Bp. 1984.

Ismerd meg a BASIC nyelvjárásait (szerk.: Kőhegyi János) Műszaki Könyvkiadó, Bp. 1984.

TV-BASIC (dr. Kocsis András) SZÁMALK, Bp. 1984.

A COMMODORE 64 mikrogép kezelése és programozása (Lángos István) COMPORGAN, Bp. 1984.

A COMMODORE VC– 20 Felhasználói kézikönyv (Dr. Úry László) LSI ATSZ, Bp. 1985.

Alapismeretek a COMMODORE 64 mikroszámítógép használatához (Halász Árpád) Novotrade, Bp.

COMMODORE 64 Felhasználói kézikönyv (szerk.: dr. Varga András) Novotrade, Bp.

ANGOL NYELVŰ IRODALOM

Commodore könyvek

VIC 20 Programmer's Reference Guide
Commodore 64 Programmer's Reference Guide
Commodore Plus/4 Programmer's Reference Guide
Mastering Your VIC 20

Four VIC 20 Computer Books:
VIC Revealed, Nick Hampshire
VIC Games, Nick Hampshire
VIC Graphics, Nick Hampshire
Stimulating Simulations for the VIC, C.W.Engel
Introduction to BASIC, Part 1 and 2, Andrew Colin
Commodore Software Encyclopedia, Third Edition

BASIC programozás

Armchair BASIC: An Absolute Beginner's Guide to Programming in BASIC, Fox & Fox, Osborne/McGraw-Hill
BASIC Handbook, Second Edition, Lien, Compusoft
Basic Commodore 64 BASIC, Coan, Hayden
Elementary BASIC, Ledgard & Singer, SRA
How to Build a Program, Emmerichs, Dilithium Press
Instant Freeze-Dried Computer Programming in BASIC, Brown
My Computer Likes Me When I Speak in BASIC, Albrecht, Dilithium Press
Nailing Jelly to a Tree, Willis, & Danley, Dilithium Press
The Programmer's Book of Rules, Ledin & Ledin, Lifetime Learning Publishers
Technical BASIC, Kassab, Prentice-Hall

Gépi nyelvű programozás

Machine Language for Beginners, Mansfield, COMPUTE! Books
Programming the 6502, Zaks, Sybex
6502 Assembly Language Programming, Leventhal, Osborne/McGraw-Hill.
6502 Micro Chart, Micro Logic Corp.
6502 Software Design, Scanlon, Sams
The 6502 Software Gourmet Guide & Cookbook, Findlay, Hayden

TÁRGYMUTATÓ

A

ablakok 43-44
animáció 59-60
ASCII kódok 147
AUTO parancs 84

B

BACKUP parancs 84
BASIC ismerettár 81-130
BASIC rövidítések és hivatkozások táblázata 131
billentyűk
- C= billentyű 19,38
- CLEAR/HOME billentyű 18,43,44
- CTRL billentyű 19
- ESC billentyű 20,44
- FLASH ON/OFF billentyűk 20,38
- funkció billentyűk 22-23
- grafikus billentyűk 21-22, 56-58
- HELP billentyű 23
- INST/DEL billentyű 18, 40-41
- írógép üzemmód 16
- kurzor billentyű 18
- RETURN billentyű 17,40
- RUN/STOP billentyű 17
- RVS ON/OFF billentyűk 19-20, 36-38
- SHIFT billentyűk 16-17
- SHIFT LOCK billentyű 17
- szín billentyűk 21

botkormány 7
BOX utasítás 67-68, 94-95

C

CHAR utasítás 66-67, 95
CHR$ kódok 147
CIRCLE utasítás 68-69, 95-96
CLEAR/HOME billentyű 18-19,43,44
CLOSE utasítás 96
CLR utasítás 96
CMD utasítás 97
COLLECT parancs 85
COLOR utasítás 61-63, 97
COPY parancs 86

D

DATA utasítás 97-98
DATASETTE magnetofon 12, 27-29
DEF FN utasítás 98
DELETE parancs 86
DIM utasítás 98-99
direkt mód 39
DIREKTORY parancs 23, 33, 86-87
DO/LOOP/WHILE/UNTIL/EXIT 99
DRAW utasítás 65-66, 100
DSAVE parancs 23, 32-33, 88

E

egész típusú változók 52-53, 127-128
END utasítás 100
ESC billentyű 20, 43,44

F

felhasználó által definiált függvények 54
fényerő 62
FOR...TO...STEP utasítás 100
fönntartott azonosítók 129
funkció billentyűk 22-23
függvények
- felhasználó által definiált 54
- karakter 125
- különleges 127
- numerikus 53, 120-125

G

GET utasítás 101
GETKEY utasítás 102
GET# utasítás 102
GOSUB utasítás 102
GOTO utasítás 38, 103
grafikus üzemmódok
- nagy felbontóképességű 63-71
- többszínű 70-71

GRAPHIC utasítás 63-64, 103

GRAPHIC CLR utasítás 104
grafika
animáció 59-60
BOX utasítás 67-68
CHAR utasítás 66-67
CIRCLE utasítás 68-69
DRAW utasítás 65-66
grafikus billentyűk használata 21, 56-58
nagy felbontóképességű 64-71
négyszögek, körök, sokszögek 67-70
többszínű 70-71
PAINT utasítás 70
pontok, vonalak, címkék 65-66
GSHAPE utasítás 116-117

H

hang
SOUND utasítás 75-76
VOL utasítás 74, 119
zenei hangjegytáblázat 143
hanghatások
zaj 77-78
zene 76-77
hatványozás 47-48
HEADER parancs 31-32, 88
HELP billentyű 23
HELP parancs 88
hibajavítás
képernyőtörlés 42-43
szerkesztés 39-42, 43-44
hibakereső táblázat 11
hibaüzenetek 30, 32-33, 136

I

IF...THEN...ELSE utasítás 104
inicializálás (lemezé) 31-32, 88
INPUT utasítás 104-105
INPUT# utasítás 105
INST/DEL billentyű 18, 40-41
inverz megjelenítés 19-20, 36-39

K

kapcsolók és csatlakozók
bottormány csatlakozók 7
kazetta csatlakozó 9
memóriabővítő kapu 6, 26
ON/OFF kapcsoló 6-7
RESET billentyű 7, 43
RF(TV) csatlakozó 8, 10
soros csatlakozó 9
tápfeszültség csatlakozó 7, 9
video csatlakozó 8, 10
karakterek beszúrása 18, 40-51
karakterek törlése 18, 40-51
kazetták 27-29
kazetta csatlakozó 8-9
kazetták töltése 27-28
kazettán való tárolás 28-29
képernyő ablakok 43-44
képernyő törlése 18-19, 42-43, 63-64
képernyőkódok 144
képernyőszerkesztés
CLR/HOME billentyű 18-19, 43, 44
ESC billentyű 20, 44
hibajavítás 39-42
INST/DEL billentyű 18, 40-41
képernyőtörlés 42-43
megjelenítés a képernyőn 50-51
KEY parancs 89
kiírás képernyőre 36-42, 50-51
kiíratási zónák 50-51
kivonás 46

L

lebegőpontos változók 52-53, 128
lemez hibaüzetek 138
lemezek 29-32
DIREKTORY parancs 23, 33, 86
inicializálás 31-32, 88
programbetöltés 29-31, 87
programtárolás 32-33, 88
LET utasítás 105
LIST parancs 23, 89-90
LOAD parancs 28, 90-91
LOCATE utasítás 106, 119-120

M

matematikai függvények származtatása 142
matematikai műveleti jelek 46-47, 130

megjelenítés képernyőn 36-42, 50-51
memóriabővítő kapu 8, 26-27
modulok
    modulok töltése 26-27
    memóriabővítő kapu 8
MONITOR utasítás 106
monitorok 10, 12
műveletek 46-47
    műveletek sorrendje 49
    zárójelek 49
műveleti jelek
    logikai 130
    matematikai 46-47, 130
    relációs 46-47, 130

N

nagy felbontóképességű grafika 63-71
NEW parancs 43, 91
NEXT utasítás 106-107
nyomtatók 13

O

ON utasítás 107
ON/OFF kapcsoló 6-7
OPEN utasítás 107-108
osztás 46

Ö

összeadás 46

P

PAINT utasítás 70, 108-109
parancsmód 39
parancsok beírása 39
perifériák 12-13
pi (π) 17, 47
pixel kurzor (PC) 119-120
POKE utasítás 109
PRINT utasítás 109-110
PRINT# utasítás 110
    PRINT USING utasítás 110-113
    programbetöltés 2
    kazettáról 27-28
    lemezről 29-31
    modulról 26-27
programmód 39
programok tárolása
    kazettán 28-29
    lemezen 32-33
PUDEF utasítás 113-114

R

rajzolás
    derékszögű négyszögek 67-68
    körök 68-69
    pontok 65-66
    sokszögek 69-70
    vonalak 65-66
READ utasítás 114
relációs műveleti jelek 46, 130
REM utasítás 114
RENAME parancs 91
RENUMBER parancs 91-92
RESTORE utasítás 114-115
RESUME utasítás 115
RETURN billentyű 17
RETURN utasítás 115
RF csatlakozó 8, 10
RUN parancs 92
RUN/STOP billentyű 17
RVS ON/OFF billentyű 19-20, 36-39

S

SAVE parancs 28-29, 32-33, 92-93
SCALE utasítás 115
SCNCLR parancs 43, 64, 116
SCRATCH parancs 93
soros csatlakozó 9
SOUND utasítás
    frekvenciatáblázat 75-76, 143
    időtartam 75-76
    szólamok 74-77
SSHAPE utasítás 116-117
STOP utasítás 118
SYS utasítás 118

SZ

számítógép összeállítása 9-11
számok
    decimális 47
    felhasználó által definiált függvények 54
    matematikai függvények származtatása 142
    műveletek 46, 48-49
    műveletek végrehajtásának sorrendje 49
    műveleti jelek 46, 130
    numerikus függvények 53
    törtek 47
    zárójelek 49
    változók 52-53, 127-129
szerkesztés (lásd a képernyő szerkesztésénél)
szín
    COLOR utasítás 61-63, 97
    fényerő 62
    inverz kijelzés 36-37, 61-63
    PAINT utasítás 70, 108-109
    színbillentyűk 21, 36-37
    színek változtatása 36-37, 61-63
    többszínű üzemmód 70-71
szoftver
    kazetta 27-29
    lemez 29-33
    modul 26-27
szorzás 46
szövegváltozók 128

T

tápegység 7-9
többszínű grafika 70-71
tömbök 128
törtek 47
TRAP utasítás 118
TROFF utasítás 119
TRON utasítás 118
tudományos jelölés 47-48
TV csatlakozókábel 6, 10

V

változók
    azonosítók 127-128
    egész 52-53, 126
    karakter 52-53, 128
    lebegőpontos numerikus 52-53, 127-128
VERIFY parancs 93-94
video csatlakozó 8, 10
VOL utasítás 74, 119

W

WAIT utasítás 119

Z

zene (lásd a hangot is) 76-77, 78-80
zenei hangjegytáblázat 143

Készült a Mozgáskorlátozottak Hajdú-Bihar megyei
Egyesülete PIREMON Kisvállalatának Nyomdaüzemében
Felelős vezető: dr. Gere Kálmán igazgató

commodore 16

"A kézikönyvnek az a célja, hogy ízelítőt adjon a számítógépről, hogy új dolgok kipróbálására serkentsen, hogy a számítógép használatára tanítson, és élvezetet nyújtson a futások során. De még annyi minden van... például a programozás... A BASIC nyelvű programozásról úgy tanulhatunk többet, hogy olyan kézikönyveket olvasunk, melyek a BASIC nyelvet tanítják. Az ilyen könyvek listája a függelékben található..."

Ára:99,– Ft